STUDY OF GIBBS DISTRIBUTED IMAGES

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Award of the Degree of

MASTER OF TECHNOLOGY

by

CHAVELI RAMESH

to the

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

April 1992

CERTIFICATE

This is to certify that this work entitled

STUDY OF GIBBS DISTRIBUTED IMAGES

by Chaveli Ramesh has been carried out

under my supervision and has not been submitted

elsewhere for the award of a degree.

Dr. P R K RAO
Professor
Department of Electrical Engineering
Indian Institute of Technology Kanpur

18 MAY 1992

CENTRAL LIBRARY

Acc. No. A.J.134.88

EE-1992-M-RAM-STU

1134-55

Acknowledgements

I would like to express my deep sense of gratitude to my thesis supervisor Dr. P R K Rao, for his valuable suggestions and guidance throughout the thesis work.

I am very thankful to Venkatesh for taking the pains to read my thesis and for suggesting the necessary corrections and also typing the difficult portions of my thesis.

I thank Y P Singh for photographing the images presented in this thesis. I thank Deepak Murthy, R Selva Kumar, Harish, Abhay Karandikar, Arun Kumar, Nandy, Visweswar, Venkaiah, Perraju, Subba Rao and Srinivas Rao for their suggestions and the constant encouragement they provided me.

Contents

Abstract

A Gibbs distributed image is one in which the probability distribution of pixel intensities is a Gibbs distribution. The distribution function is characterised by a set of parameters. In this work, an image generation method known as the spin-flip algorithm has been used to generate Gibbs distributed images with varying sets of parameters and the effect on the texture of the image of changing various parameters has been studied. Also, an attempt has been made to come up with a parameters estimation method based on histogramming. The method has been tested on various simulated images. Further, an image compression method has been proposed that operates by modelling images as Gibbs distributions. A compression ratio of two has been obtained for various simulated images. A texture classification method has been suggested that depends upon minimising the Euclidian norm from the parameter set of the image to the parameter set of each class. Satisfactory results have been obtained.

Chapter 1

Introduction

1.1 MOTIVATION

The recognition that a class of images could be realistically described by Gibbs distribution opened up interesting possibilities for processing of images.

A Gibbs distributed image is one in which the probability distribution of the pixel intensities is a Gibbs distribution. Gibbs distributions were initially used to model molecular interactions in ferromagnetic materials. Later, they were used as models for binary alloys, lattice gas and for agricultural studies.[13]

In the past two decades, interest in the use of the Gibbs distribution has intensified in the field of image processing. The equivalence of Gibbs distributions and of Markov random fields that was proved in the first half of the 1970s eliminated certain difficulties in dealing with conditional distributions in Markov random fields [14], [15]. In the use of the Markov random field, it was necessary to explicitly state the form of certain joint distributions; the need for this was eliminated when the equivalent Gibbs distribution was used instead. It was

primarily the increased facility resulting from the possibility of substituting an equivalent Gibbs distribution formulation that did not entail the complexities inherent in a Markov random field model that encouraged the extensive use of the Gibbs distribution in image processing problems.

Various algorithms [19], [20] have been devised for generating Gibbs distributed images and estimating their parameters. While for generating Gibbs distributed images, satisfactory algorithms such as the spin-filp and the Metropolis algorithm were invented, (these methods have been extensively studied in the physics literature), for the estimation of parameters, there are not many well established methods. Although Besag's method [13] for estimating the parameters is known to give good results under certain conditions, it does not always prove successful [13]. Also, the use of the models in various applicatioms such as image compression, image interpolation, image classification, etc. is yet to be thoroughly studied.

In this thesis, we study various aspects of the spin flip algorithm by using the algorithm to generate Gibbs distributed images. Also, a parameter estimation method based on histogramming has been presented. Next, we explore the application of the Gibbs distribution to the problems of image compression and texture classification.

1.2 MODELLING OF IMAGES

Images can be represented in many ways, such as as a rectangular lattice of point sites with the intensity at each point taking values from a countable set, or as a

rectangular lattice of point sites with the intensity at each point taking values from an uncountable set. In general, the intensities in an image might appear to take values in a continuous space, but dealing with continuous variables is computationally inconvenient; so, we represent images as rectangular lattices of point sites with the values of the points lying in a discrete space. From now on, we call a lattice point a *pixel*. Mathematically, an image is a two dimensional array with each element in the array taking values in discrete space.

The purpose of modelling an image is to capture the inherent regularities in an image. We can model an image as one in which each pixel intensity is some function of the neighbourhood pixels' intensities and of the pixel's location in the image. This approach leads us to a deterministic way of modelling an image. A different approach which is often used is to model the image as a random field. We follow the latter method.

Initially, there were two approaches to modelling images as spatial stochastic processes. In the following discussion, we label image pixels by (i,j) and with each pixel we associate a random variable $X[i,j]$. The first approach requires that the joint distribution $P(X[1,1]=x_{1,1},\ X[1,2]=x_{1,2},\ \ldots,\ X[N,N]=x_{N,N})$ should be in product form, $\prod_{i,j=1,1}^{N,N} Q_{i,j}(x_{i,j};\eta_{i,j})$ with each term in the product having the form $Q_{i,j}(x_{i,j};\eta_{i,j})$, where $x_{i,j}$ is the value of the random variable $X[i,j]$ and $\eta_{i,j}$ is the neighbourhood (the neighbourhood of a pixel is a set of pixels that are associated with it by a rule) of pixel (i,j). The second approach suggests that the probability distribution of $X[i,j]$ conditioned over all site intensity values except of (i,j) itself should depend only upon some neighbourhood site intensity values. It has been established through the Hammersely—Clifford theorem [13] that both these approaches are equivalent. Since the Markov random field model is a conditional

probability model and the Gibbs distribution model is a joint distribution model, the Hammersely— Clifford theorem effectively establishes the equivalence of Markov random fields and Gibbs ditributions.

Every model is characterised by its parameters. Once the model has been selected, the estimation of parameters is an important problem. There can be two types of situations: one is where many realizations of images are given and we need to estimate the parameters for the single model that governs the behaviour of all the images; the other is where only a single image realization is given and it is required to estimate the parameters from that image alone (this situation calls for certain assumptions of ergodic behaviour, as what is being attempted is to estimate the parameters of the distribution from the single available realization). In the former case, the parameters can be allowed to vary even from pixel to pixel but in the latter case, the parameters would have to remain same for sufficiently large regions of the image, in order that the parameters may be estimated accurately.

In a Markov random field, the probability distribution of any pixel is dependent upon the (intensities at the) neighbourhood pixels. The dependence can be assumed to be of a causal kind or of a non-causal kind. In a causal model, the probability distribution of $X[i,j]$ is conditional only upon $X[k,l]$ for all (k,l) in a subset of the region formed by $k<i$ and $l<j$, whereas in a non-causal model, the probability distribution of $X[i,j]$ is dependent on $X[k,l]$, for all (k,l) in a subset of the entire lattice, merely excluding the pixel (i,j) itself. We use only the non-causal models because they are more realistic in the context of images.

If the probability distribution of $X[i,j]$ for all (i,j) is modelled as being dependent upon the nearest four pixels, it is called a first order model. If, instead, the distribution is modelled as dependent upon the nearest eight neighbuors, it is called a second order model. In a similar fashion, still higher orders can be defined. As the order increases, the parameters also increase in number; thus, dealing with higher order models is computationally complex.

As already mentioned, for the generation of Gibbs distributed images, there are well established algorithms—these are the Metropolis algorithm and the spin-flip algorithm. In both these algorithms we start from an arbitrary image and then perform a sequence of state changes such that the resulting image distribution is the required Gibbs distribution. The state changes can be carried out in an asynchronous or in a synchronous fashion. In a synchronous type of updating, all pixels change state simultaneously. In an asynchronous type of updating, one pixel is changed at a time and with the change introduced, the next pixel is updated and so on. Pixel selection can also be done in a deterministic or a random fashion. In case of asynchronous updating the only condition that should be satisfied while selecting the pixels is that each pixel should be visited equally often in the mean.

1.3 APPLICATION OF GIBBS DISTRIBUTED IMAGES

The use of the Gibbs distribution in image processing is fairly recent, but the potential for its use in statistical image models is enormous. Most of the work until now has been concentrated to devise image segmentation algorithms. Initially, segmentation algorithms requiring many passes [22] were designed but recently, segmentation algorithms requiring only a single pass [22] have appeared. Adptive

segmentation algorithms [23] which simultaneously estimate the parameters of the Gibbs random field and segment the noisy images corrupted by additive independent Gaussian noise have also been devised. Gibbs models also find applications in modelling fractals [25]. Algorithms for yielding binary fractals by modelling images as Markov random fields have also been devised [25].

Recently, reports of the use of Gibbs models in the field of texture classification have also appeared [27]. These models have also been used to model carpets and to identify defective textile textures [32]. Classification of rotated and scaled textures can also be done using these methods [27].

Applications of Gibbs models in the area of image compression, interpolation etc., to the author's knowledge, have not been reported in the literature. In this thesis, an attempt has been made to apply Gibbs models in the field of image compression and texture classification.

1.4 ORGANISATION OF THE THESIS

In this thesis we have investigated various aspects relating to the generation of the Gibbs distributed images. Convergence properties of the spin-flip image generation algorithm for cases with and without annealing have been investigated and effects of changing various parameters have also been studied. Also proposed is a parameter estimation method based on histogramming the image. Next, the Gibbs distribution models are applied for image compression and texture classification. All the ideas have been tested on simulated images.

In Chapter 2, we briefly discuss the previous work on Markov random fields and Gibbs distributions. In Chapter 3, we discuss the various aspects of an image generating algorithm, the spin-flip algorithm. In the same chapter, we also discuss a parameter estimation method. Using the method, parameters of various simulated images are estimated: then, using the estimated parameters, images are again generated. The textures of the initial and final images are compared. In Chapter 4, the application of Gibbs models to image compression and texture classification is discussed. Various examples supporting the ideas have been presented. In Chapter 5, we present the conclusions and some suggestions for future work.

Chapter 2

Review of the Relevant Literature

The application of statistical models, particularly of the Markov random field models, to image processing has increased considerably in recent years. In this chapter, we present the definitions and issues related to Markov random fields.

2.1 MARKOV RANDOM FIELDS IN IMAGE PROCESSING

There is a vast body of literature on various spatial interaction models and image restoration and parameter estimation algorithms based on Markov random field models. In their work on Markov random fields (MRFs) Dobrushin [1], Wong [2] and Woods [3] extended the Markovian property from one dimension to higher dimensions. However, the computational complexity of the algorithm in two dimensions necessitates restrictions on the class of MRF models. The model for images is restricted to a special class of MRFs called Markov mesh random fields (MMRFs) which are characterised by causal transition distributions. This approach has been proposed by Abend et al [4]. Woods and Radevan [5] also use MMRFs for

image processing applications. Other attempts in extending the Markovian property to two dimensions include include autoregressive models: the "simultaneous auto regressive models" (SAR models) and the "conditional Markov" models introduced by Chellappa and Kashyap [6].

There are various applications of Markov random fields and two dimensional autoregressive models in image segmentation. MRF models, Bayesian approaches and MAP formulations are combined in the work by Kaufman et al [7] with reduced update Kalman filtering techniques. Therrien [8] uses two-dimensional autoregressive texture models for texture based segmentation. Hansen and Elliott [9] also suggest a segmentation method using MRF models and MAP formulation. However, until the equivalence of MRFs and Gibbs distribution was discovered, the full power of MRFs to capture spatial interactions could not be exploited.

The Gibbs distribution (GD), introduced by Ising [10] in 1925 was used to model molecular interaction in ferromagnetic materials. Gibbs distributions have also been used as models for lattice gas [11], binary alloys [12], and for agricultural studies in the pioneering work by Besag [13]. The Gibbs distribution also has applications in neural modelling of inference and learning, social and economic models, optimal VLSI design applications and image processing applications.

The eqivalence of GDs and MRFs was proved independently by Avertizev [14] and Spitzer [15]. The celebrated work by Besag [13] laid the ground work for GD characterization of MRFs. Kinderman and Snell [16] also present a detailed study of GD-MRF equivalence. The GDs and MRFs eliminated the difficulties involved in dealing with conditional distributions in MRFs, because the joint distributions are readily available in the form of the Gibbs distribution function. Hassner and

Skalansky [17] used Gibbs distributions to model textures. They used a first order binary MRF, the Ising model, to characterise binary textures; they also presented an algorithm for generating textures and for finding the paramters. Cross and Jain [18] generalised it for the M-ary case. They used Besag's coding method to estimate parameters and the Metropolis algorithm [19] to generate the texture. In the work by Geman and Geman [20], the Gibbs distribution characterisation of MRFs is used to develop image segmentation methods.

Cohen and Cooper [21] also used GDs for image segmentation. Derin, H and co-workers [22], present GD models for noisy and textured images. Lakshmanan and Derin [23] propose an adaptive segmentation algorithm (ASA), which simultaneously estimates the parameters of the underlying Gibbs random field (GRF) and segments a noisy image corrupted by additive independent Gaussian noise. Geiger, D and Girosi, F [24] derive a deterministic approximation to MRFs. This model is applied to smoothen an image preserving its discontinuities. Onural, L [25] proposes a procedure to yield textures and connected binary fractals. Ari Veijanen [26] proposed a new estimator for estimating the parameters of an MRF from noisy observations. Cohen et al [27] present a Gaussian MRF-based rotated and scaled texture classification method.

2.2 *MARKOV RANDOM FIELDS*

We consider a two dimensional lattice of points L defined as

$$L = \{(i,j) |\ 1 \le i \le N_1,\ 1 \le j \le N_2\}.$$

On this lattice, we define an image random field X as

$$X = \{X(i,j) |\ 1 \le i \le N_1,\ 1 \le j \le N_2\}.$$

For our purpose, the points of the lattice correspond to the pixels of a digitised image.

DEFINITION 2.1 *Neighbourhood: The neighbourhood η_{ij} of pixel (i,j) satisfies the following conditions.*

(a) $(i, j) \notin \eta_{ij}; \quad (i, j) \in L$

(b) $(k, l) \in \eta_{ij} \Rightarrow (i, j) \in \eta_{kl}; \quad (i, j) \in L$ ...1

DEFINITION 2.2 *Neighbourhood System: A neighbourhood system is defined as*

$$\eta = \{\eta_{ij} \mid \eta_{ij} \subset L;\ (i, j) \in L\} \qquad ...2$$

DEFINITION 2.3 *Markov Random Field: Let L be a lattice and η be a neighbourhood system defined over L. A random field $X = \{X[i,j],(i,j)\in L\}$ is a Markov random field (MRF) with respect to the neighbourhood system η if and only if*

$$P(X(i, j) = x_{ij} / X(k, l) = x_{kl};\ (k, l) \in L,\ (k, l) \neq (i, j))$$

$$= P(X(i, j) = x_{ij} / X(k, l) = x_{kl};\ (k, l) \in \eta_{ij}); \quad (i,j) \in L. \qquad ...3$$

By taking large neighbourhoods, elaborate spatial dependences can be modelled. But for most images of interest, simple neighbourhoods are adequate. A hierarchy of neighbourhood systems that are used in image processing is $\eta^1, \eta^2, \eta^3, \ldots$ $\eta^1 = (\eta^1_{ij})$ where η^1_{ij} consists of the nearest four pixels of each $(i,j) \in L$. Similarly $\eta^2 = (\eta^2_{ij})$ where η^2_{ij} consists of the nearest eight pixels of each (i,j) and so on. Fig 2.1 shows $\eta^1_{ij}, \eta^2_{ij}, \eta^3_{ij}, \eta^4_{ij}$.

The conditional distributions in equation (3) are called the local charactristics of the random field. This characterization is intutively appealing because it is natural to expect that the intensity of a pixel depends on its neighbouring pixels rather than upon those outside its neighbourhod.

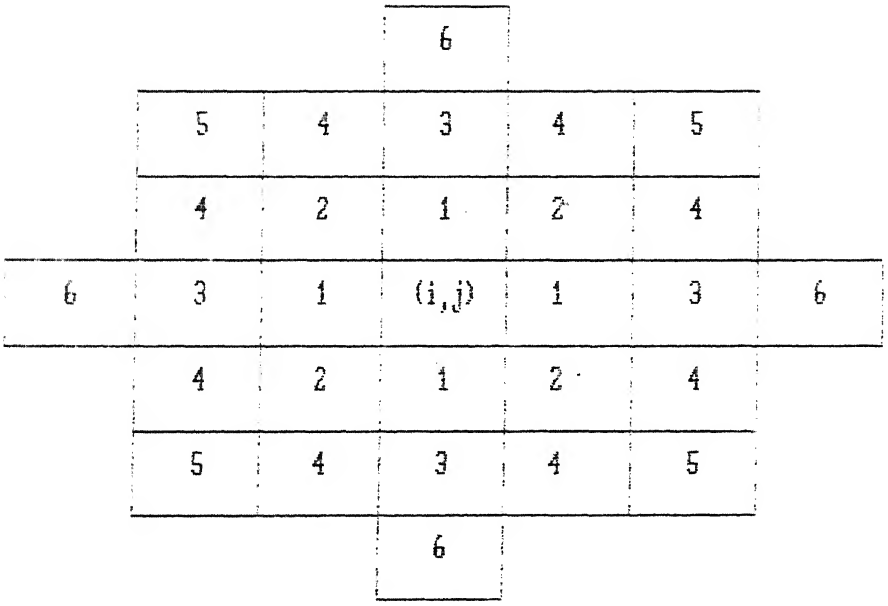


Figure 2.1: η^{1}_{ij}, η^{2}_{ij}, η^{3}_{ij}, η^{4}_{ij}, η^{5}_{ij}

Every Markov random field with $P(X=x)>0; \forall x$ is also a Gibbs distribution and vice versa. This equivalence makes it much simpler to write down the joint distribution in an explicit form. The condition that $P(X=x)>0; \forall x$ is a requirement for proving the Hammersley—Clliford theorem, which states that an MRF with $P(X=x)>0; \forall x$ has the same probability structure as a Gibbs distribution and vice versa. A detailed account can be found in Besag [13].

2.3 GIBBS DISTRIBUTIONS

Before defining a Gibbs distribution we have to define a clique. Let us first define the cliques associated with (L, η), a lattice-neighbourhood system pair.

DEFINITION 3: *A clique of the pair* (L, η), *denoted by* c, is a subset of L *such that*

a) c *consists of a single pixel, or*

b) $(i, j) \neq (k, l)$ *and* $(i, j) \in c$ *and* $(k, l) \in c \Rightarrow (i, j) \in \eta_{kl}$

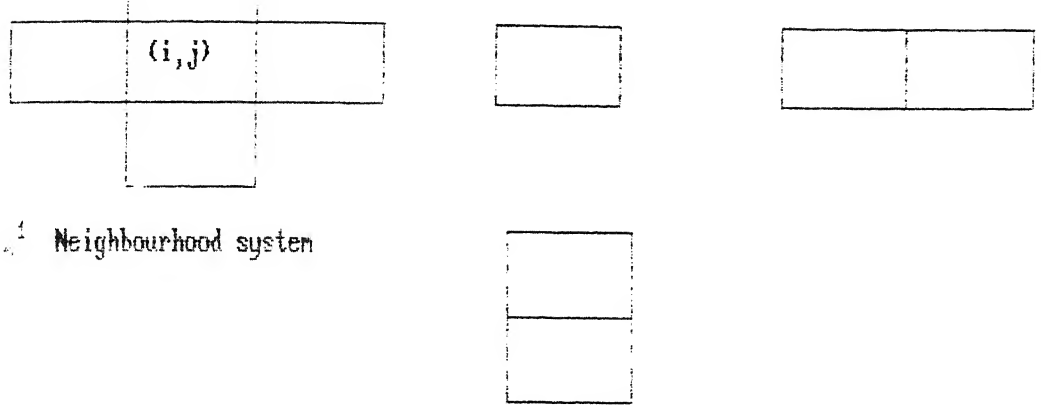
The collection of all cliques of (L,η) *is denoted by* $C(L,\eta)$. *The cliques associated with* η^1 *and* η^2 *are shown in Fig 2.2. We now define a Gibbs random field (GRF).*

GIBBS RANDOM FIELD: *Let* η *be a neighbourhood system defined over a lattice* L. *A random field* $X=\{x_{ij}\}$ *defined on* L *is a Gibbs random field (GRF) (or a random field having a Gibbs distribution) with respect to the neighbourhood system* η *if and only if its joint distribution is of the form*

$$P(X=x) = \frac{1}{Z} \cdot e^{-U(x)} \qquad \ldots 4$$

where

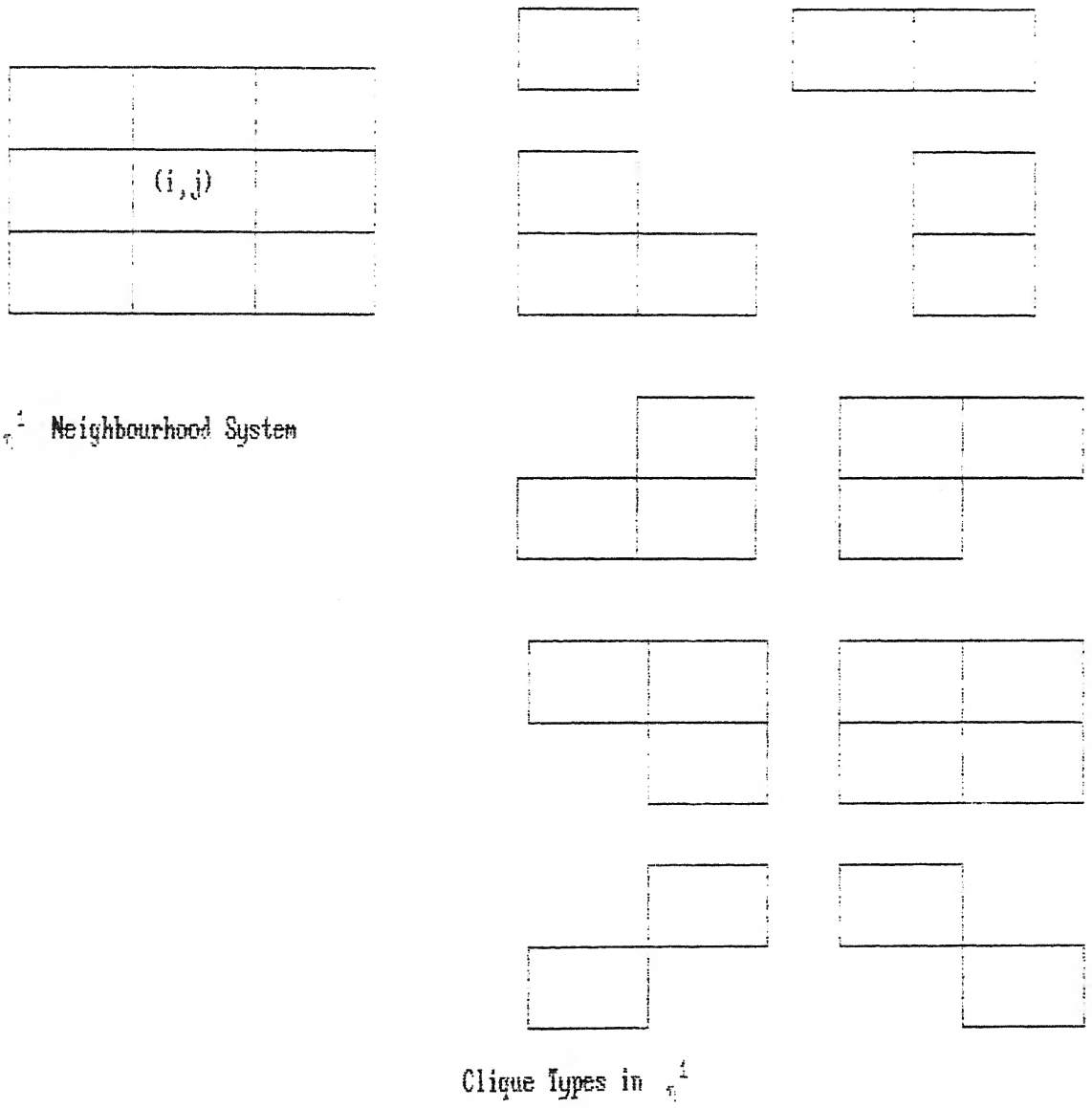
$$U(x) = \sum_{c \in C} V_c(x) \quad \triangleq \quad \textit{energy function.}$$



$\mathcal{N}_1$ Neighbourhood system

Clique Types in $\mathcal{N}_1$

(a)



$\mathcal{N}_1$ Neighbourhood System

Clique Types in $\mathcal{N}_1$

(b)

Fig 2.2: Neighbourhood Systems $\mathcal{N}_1$ and $\mathcal{N}_2$

and

$$Z = \sum_{x} e^{-U(x)} \triangleq \textit{partition function.}$$

Z is a normalising constant, so that the sum of the probabilities of all realisations is unity. The clique potential $V_c(x)$ is any arbitrary function, with the only necessary condition that it should be a function of the pixel values in the clique. Ising [10] first used a special type of GD to model a binary nearest neighbour system.

The joint distribution function in equation (4) has the interpretation that the smaller the energy $U(x)$, of a realisation of the state x becomes, the more probable is the realisation of that state. In studies related to physical systems, a temperature term is introduced in the exponent of the joint distribution and the exponent is expressed as $-U'(x)/T$ where T is the temperature and $U'(x)$ is the energy function. The interpretation of the effect of T is that when T is large then a great many realizations are possible.

A temperature parameter is useful if the state of the 'system' is desired to converge to a minimum energy state by passing through state transitions from an arbitrary initial state. If the temperature T is decreased according to a given schedule so that the state of the system gets caught among highly probable states, then convergence to the lowest energy state is assured. This process is called annealing. But if the temperature is decreased too fast, the system can get locked in a local mnimum energy state and the global minimum energy state may never be reached. The schedule according to which the temperature is to be decreased is called the annealing schedule [20]. The fastest rate of temperature decrease that may be permitted is an important consideration.

We now state the Hammersley-Clifford theorem that describes the one-one correspondence between MRFs and GDs. The proof can be found in [13], [16].

THEOREM-1 *Let η be a neighbourhood system defined on a finite lattice L. A random field $X=\{x_{ij}\}$ is a Markov random field with respect to η if and only if its joint distribution is a Gibbs distribution.*

In proving this theorem, $P(X=x)>0; \forall x$ is an essential condition. It follows from this theorem that MRF satisfies the following distribution function:

$$P(X(i,\ j)=x_{ij}/X(k,\ l)=x_{kl};\ (k,\ l)\in\eta_{ij}) \ = \ \frac{exp\left(\ -\sum_{c\in C} V_c(x)\right)}{\sum_{x_{ij}} exp\left(\ -\sum_{c\in C} V_c(x)\right)} \qquad ...5$$

2.4 *A CLASS OF VALID GIBBS DISTRIBUTIONS*

The Hammersley—Clifford theorem addresses the question of the general form of the energy function $U(x)$ that must be maintained so as to ensure a valid probability structure of an MRF. The theorem states that any function of the following form is valid.

$$U(x) = \sum_{[i_1,j_1]} x_{i_1j_1} G_{i_1j_1}(x_{i_1j_1}) + \sum_{\substack{[i_1,j_1] \\ [i_2,j_2]}} x_{i_1j_1}x_{i_2j_2} G_{i_1j_1,i_2j_2}(x_{i_1j_1},\ x_{i_2j_2}) + \cdots$$

This may be expressed as

$$U(x) = \sum_{n=1}^{N_1 \cdot N_2} \sum_{\substack{[i_1,j_1] \\ \vdots \\ [i_n,j_n]}} x_{i_1j_1}\cdots x_{i_nj_n} \cdot G_{i_1j_1,\ldots,i_nj_n}(x_{i_1j_1},\ \ldots,\ x_{i_nj_n}). \qquad ...6$$

where $G_{i_1j_1,\ldots,i_nj_n}$ is a function of $(x_{i_1j_1},\ \ldots,\ x_{i_nj_n})$. Within the general framework of equation (6), we can make particular choices depending on their suitability to the

practical situation we wish to deal with. For example, if $U(x)$ is defined in the following form,

$$U(x) = \sum_{[i_1,j_1]} x_{i_1j_1} G_{i_1j_1}(x_{i_1j_1}) + \sum_{\substack{[i_1,j_1] \\ [i_2,j_2]}} x_{i_1j_1} x_{i_2j_2} G_{i_1j_1,i_2j_2}(x_{i_1j_1}, x_{i_2j_2})$$

the models obtained are called auto-models.

AUTO MODELS: Besag has described the following classes of auto models [13].

1. Auto-logistic: For a binary random field, if the energy function is of the following form

$$U(x) = \sum_{[i_1,j_1]} x_{i_1j_1} \alpha_{i_1j_1} + \sum_{\substack{[i_1,j_1] \\ [i_2,j_2]}} x_{i_1j_1} x_{i_2j_2} \beta_{i_1j_1,i_2j_2}$$

where $\alpha_{i_1j_1}$ and $\beta_{i_1j_1,i_2j_2}$ are constants with respect to site locations but depend upon the clique configuration, the model is called an auto logistic model. Auto logistic models have been used by Flinn [12], Cohen and Cooper [21] and Hassner and Skalansky [17].

2. Auto-normal models: If the set of random variables that constitute the random field are jointly Gaussian, it can be seen that the quadratic in the exponent of their joint distribution is of the form of equation (6). Such a model is called an auto normal model [13].

3. Auto-Poisson model: The auto-Poisson model introduced by Besag [13], is defined such that each random variable in the random field is conditionally Poisson distributed with parameters depending upon the neighbours of that random variable.

4. Derin [30] proposes yet another kind of auto-model which is versatile and powerful. Suppose that each of the random variables $X[i_1,j_1]$ takes one of the M values m from the set $\{0, 1, \ldots, M-1\}$. For this class of models $G_{i_1j_1}(x_{i_1j_1})$

and $G_{i_1j_1,i_2j_2}(x_{i_1j_1}, x_{i_2j_2})$ have the following form.

$$G_{i_1j_1}(x_{i_1j_1}) = \alpha_m \quad if \quad x_{i_1j_1}=m, \ \forall [i_1, j_1] \in L$$

$$G_{i_1j_1,i_2j_2}(x_{i_1j_1}, x_{i_2j_2}) = \beta_{i_1j_1,i_2j_2}|x_{i_1j_1}-x_{i_2j_2}|^r \quad if \ x_{i_1j_1}=m, \ x_{i_2j_2}=n$$

where m and n are the possible levels the pixel intensity can occupy. The parameter α_m serves as the external field for the spatial interaction system and it determines the number of pixels of each level in the image. The parameter $\beta_{i_1j_1,i_2j_2}$ decides the penality or reward for equality of $x_{i_1j_1}$ and $x_{i_2j_2}$. The parameter r also affects the penalty or reward.

2.5 *PARAMETER ESTIMATION METHODS*

Given a single image realisation and a particular model, the determination of parameters of that model constitutes the estimation problem. It is assumed in the present work that the parameters remain invariant to location in the lattice .

1. Coding method: Let us consider a rectangular lattice L and a single realisation X. We want to estimate the parameters of the assumed model distrbution using the single realisation. In order to estimate the parameters of the first order distribution model (*i.e.*, the GD model with a nieghbourhood system of the first order), we begin by labelling the interior sites of the lattice alternately by ∘ and • as shown in figure 2.3. It is clear from the first order Markovian assumption that the variables associated with ∘ sites given variables at the • sites, are mutually independent. So the conditional likelihood function over the ∘ sites can be written in the following form.

$$\prod_{[i,j]\in(\circ)} P(X(i, j)=x_{i,j}/X(i-1, j)=x_{i-1,j}, \ X(i, j+1)=x_{i,j+1}, \ X(i+1, j)=x_{i+1,j}, \ X(i, j-1)=x_{i,j-1})$$

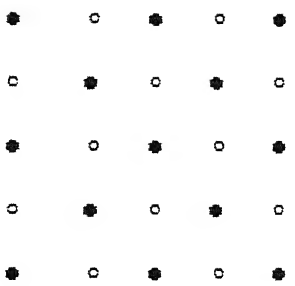


Figure 2.3: Besag's Coding on a Rectangular Lattice

v_1	u_2	v_2
u_1	s	u_3
v_4	u_4	v_3

Figure 2.4: u_is and v_is With Respect to s

where the product is over values associated with ∘ sites. Conditional likeihood estimates of the unknown parameters can then be obtained. Alternatively, the parameters can be obtained by using the conditional likelihood function for the • sites. The two estimates ought to turn out to be the same but it is advisable to combine both the estimates. Similarly, model parameters can be estimated for higher order neighbourhod systems by this method. This method is computationally efficient.

2. The method of Derin and Elliot: Histogramming and standard least square estimation are the main components in this method. Suppose X is a GRF with each $X[i,j] \in \{0, 1, \ldots, M-1\}$. Let s represent the pixel (i,j) and t' represent the vector of the neighbouring pixels of (i,j). $t' = [u_1\ u_2\ u_3\ u_4\ v_1\ v_2\ v_3\ v_4]^T$ where the u_is and v_is are shown with respect to s in figure 2.4. We define the indicator functions

$$I(z_1, z_2, \ldots, z_k) = \begin{cases} -1; & z_1 = z_2 = \ldots = z_k \\ 1; & otherwise \end{cases}$$

where $z_1, z_2, \ldots, z_k$ are the pixel intensity values.

$$J_k(s) = \begin{cases} 1; & s = m \\ 0; & otherwise \end{cases}$$

We now express the potential functions of the GDs in terms of these quantities. Let $V(s, t', \theta)$ be the sum of the potential functions of all cliques that contain (i, j), the site of s. That is, $V(s, t', \theta) = \sum_{c:s \in c} V_c(x)$. We define θ to be a parameter vector such that $\theta = [\alpha_1, \alpha_2, \ldots, \alpha_M, \beta_1, \ldots, \beta_4, \gamma_1, \ldots, \gamma_4]$. The formulation considered here is in terms of the η^2 neighbourhood system. Using the clique potentials for this class of GD, we can write $V(s, t', \theta)$ as

$$V(s, t', \theta) = \phi^T(s, t') \cdot \theta$$

$\phi^T(s, t')$ is a function of the various indicator functions. Now, suppose $P(s,t')$ is the joint distribution of the random variables on the 3X3 block centred at (i, j) and $P(t')$ is the joint distribution of the pixels in $\eta_{i,j}$ only. Then

$$P(s/t') = \frac{P(s,t')}{P(t')} = \frac{1}{Z}\cdot e^{-V(s,t',\theta)}$$

This can be rearranged to

$$\frac{Z}{P(t')} = \frac{e^{-V(s,t',\theta)}}{P(s,t')}.$$

The left side is independent of s and so is the right side. Taking the ratio for $s = j$ and k, we get

$$e^{-V(j,t',\theta)+V(k,t',\theta)} = \frac{P(j,t')}{P(k,t')}$$

Hence,

$$V(k,\ t',\ \theta) - V(j,\ t',\ \theta) \ = \ (\phi^T(k,\ t') - \phi^T(j,\ t'))\cdot\theta \ = \ ln\frac{P(j,t')}{P(k,t')}$$

Hence, we get a linear set of equations with θ as an unknown parameter set. $P(s,t')$ is calculated using histogram techniques.

2.6 *GENERATION OF MARKOV RANDOM FIELD MODELED IMAGES*

There are mainly two methods for generating an image realization from the MRF. These are the following.

METROPOLIS ALGORITHM: The Metropolis algorithm was devised to study equillibrium properties, especially ensemble averages, time evolutions, and low temperature behaviours of very large systems of essentially identical interacting components such as molecules in a gas or atoms in binary alloys. In this algorithm, two pixels are first chosen at random. Their values are exchanged if they are different and if the exchange takes the system into a more probable (i.e., lower energy) configuration. If, on the other hand, the new configuration is *less* probable than the earlier one, an exchange will be effected with a probability equal to the ratio of the probabilities of the new and the old configurations (that this is indeed a legitimate probability function is ensured by the fact that this procedure is

resorted to only when the newer configuration has a lower probability than the previous one). The procedure is practically implemented by comparing samples of a random variable uniformly distributed over [0,1] with the ratio and effecting the exchange only whenever the sample value is less than the ratio.

This randomization allows state changes to less probable conafigurations also. This is precisely what prevents the system from remaining in some local energy minima. The ratio of the probabilities of the new and the old configuration can be easily calculated due to the equivalence of GDs and MRFs, without actually determining the probabilities themselves. It has been proved that this algorithm will converge to a configuration that maximizes the joint probability, but determination of the rate of convergence is a difficult problem and a satisfactory solution does not exist. The initial configuration does not affect the convergence properties of the algorithm. It might at most take more or fewer itertions depending upon the the initial configuration. An important point to note is that the total number of pixels of each value remains the same in the case of Metropolis algorithm. This might be a drawback. Cross and Jain have used the algorithm to generate textures.

GIBBS SAMPLER AND THE SPIN-FLIP ALGORITHM: Geman and Geman [20] have used a differnt type of algorithm to generate textures. On each scan, a pixel is picked at random or in a deterministic fashion. Then the pixel value is changed to one of the values it can take, with a conditional distribution of

$$P(X(i,\ j) = x_{i,j} / X(k,\ l) = x_{k,l};\ (k,\ l) \in \eta_{i,j}); \qquad (i,j) \in L.$$

The pixel visiting mechanism can be deterministic or random but must be such that each pixel is visited equally (infinitely) often as the number of iterations increases to infinity. Geman and Geman call this algorithm a Gibbs sampler.

The annealing temperature in the probability distribution is reduced as the algorithm proceeds. They have proved that with an annealing schedule given by: $T(k) \geq c_0 / log(1+k)$. where c_0 is any constant and k the number of iterations, the configuration generated by the algorithm will have a Gibbs distribution and have minimum energy with probability converging to one as $k \rightarrow \infty$. But this annealing schedule is very slow.

Chapter 3

Generation and Parameter Estimation for Gibbs Distributed Images

3.1 THE MODEL

Many types of Gibbs models are found in the literature. Depending upon the choice of energy function, we get different types of Gibbs models. Once the model is chosen, the parameters governing the model can be varied and images with different parameters can be realised. While images can be binary or m-ary, we deal here with binary images only. The model we have chosen is a specific type of auto Gibbs model.

Let $x[i,j]$ be a $N \times N$ array representing the image. We shall consider a five parameter model. Of these, four parameters effect the spatial interaction and one parameter affects the number of white pixels in the image. These features may be illustrated using examples. The energy function used for simulating the images is the following.

$$U(x[i,j],\eta_{ij}) = L.\frac{(x[i,j]+1)}{2}$$

$$+ b_1\{(x[i-1,j-1]-x[i,j])^2 + (x[i+1,j+1]-x[i,j])^2\}$$

$$+ b_2\{(x[i-1,j]-x[i,j])^2 + (x[i+1,j]-x[i,j])^2\}$$

$$+ b_3\{(x[i-1,j+1]-x[i,j])^2 + (x[i+1,j-1]-x[i,j])^2\}$$

$$+ b_4\{(x[i,j+1]-x[i,j])^2 + (x[i,j-1]-x[i,j])^2\}$$

$U(x[i,j],\eta_{ij})$ is the energy associated with the (i,j)th pixel having a neighbourhood η_{ij}.

3.2 *THE TEXTURE GENERATION ALGORITHM*

The algorithm used here is the spin-flip algorithm and is specifically described here for binary images. We call one run of the algorithm one iteration. Each iteration consists of the following steps. Pixels are picked either in order in a raster fashion, or at random and the probabilities of the two values the pixel intensity can take are computed using the following GD equation.

$$P(X[i,j]=x_{ij}) = \frac{exp\left[-\frac{U(x_{ij},\eta_{ij})}{T}\right]}{\sum_{x_{ij}} exp\left[-\frac{U(x_{ij},\eta_{ij})}{T}\right]} \qquad ...3.1$$

where T is a parameter which should be varied in the following manner (k is the number of iterations).

$$T \geq 3.0/ln(1+k) \qquad ...3.2$$

If the probability of the pixel having the present value is higher than the probability of the pixel taking the other possible value, the pixel value is left unaltered in the present state. But if the probability of the present value is less than the probability of the other value, a transition to the other value is made

with a probability equal to the ratio of the probability of present value to the probability of the other value. For this purpose, a sample of a random variable uniformly distributed in [0,1] is compared with the ratio and the transition to the other value made only if the random number is greater than the ratio. This algorithm is used with $T=1$. The parameter T is then varied using the expression 3.2 under equality. Reducing T as in expression 3.2 is called annealing. The algorithm may be summerised as follows:

INPUT: A NXN BINARY IMAGE FILE, PARAMETERS.

OUTPUT: GIBBS DISTRIBUTED IMAGE WITH CHOSEN PARAMETERS.

STEP-1: $i=1, j=1$

STEP-2: Find $U(x[i,j],\eta_{i,j})$

STEP-3: Find $P(X[i, j]=-1)$, $P(X[i, j]=1)$

STEP-4: Let the initial value of the pixel be x_0 and the latter value be y_0, *if* $P(X[i, j]=x_0) \geq P(X[i, j]=y_0)$, make no change of value.

STEP-5: *If* $P(X[i, j]=x_0) < P(X[i, j]=y_0)$, generate a realization of Z_0 with probability function U[0,1],

STEP-6: *If* $z_0 < P(X[i, j]=x_0)/P(X[i, j]=y_0)$, make no change, *else* change value to y_0.

STEP-7: $j=j+1$,

STEP-8: *If* $j=N+1$, *then* $i=i+1$ and $j=1$

STEP-9: *If* $i<N+1$, *then goto* STEP-2 *else* stop.

3.3 CONVERGENCE PROPERTIES OF THE ALGORITHM

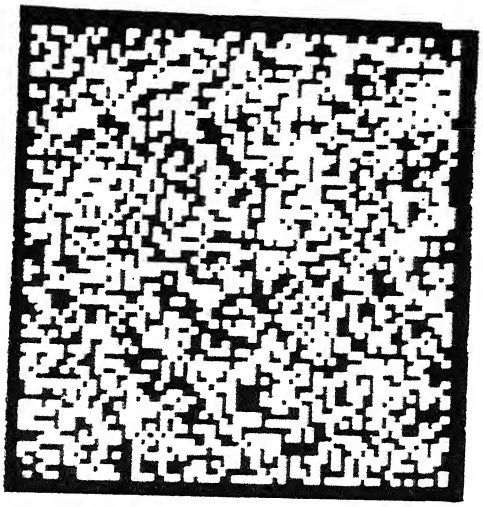
The algorithm is used for generating images with and without annealing. Four examples of each case are presented with changing sets of parameters.

IMAGE GENERATION WITHOUT ANNEALING: For different starting images, the texture of the stabilised image remains the same for the same set of parameters. But if the parameters are varied, the texture changes. The images are created by applying the algorithm for 50 iterations because it is observed that after 20 iterations, the visual features of the image stabilise. Throughout this, T is fixed at unity.

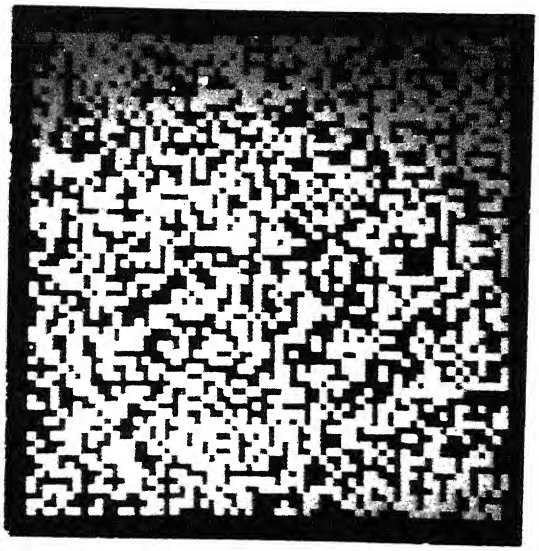
Fig-3.1(a) and Fig-3.1(b) show images idata and idata1 starting from which the Gibbs distributed images are generated. These images are generated by using a random number generater which generates random numbers uniformly distributed between 0 and 1. For each pixel, a uniformly distributed random number between 0 and 1 is called, and if the random number is less than 0.5, a value of $+1$ is assigned to the pixel; otherwise a value of -1 is assigned.

Fig-3.2(a) shows the stabilized image obtained after applying the algorithm for 50 iterations starting from idata. Fig-3.2(b) shows the stabilized image (also after 50 iterations) starting from idata1 and Fig-3.2(c) shows difference. So, the stabilized image textures resemble one another though the images are different when the starting images are different. (see Appendix 1 & 2)

IMAGE GENERATION WITH ANNEALING: In tha case of image generation with annealing T is reduced with each iteration. So, it may be expected that fewer pixel changes occur as the number of iterations increases. This is due to the nonlinear

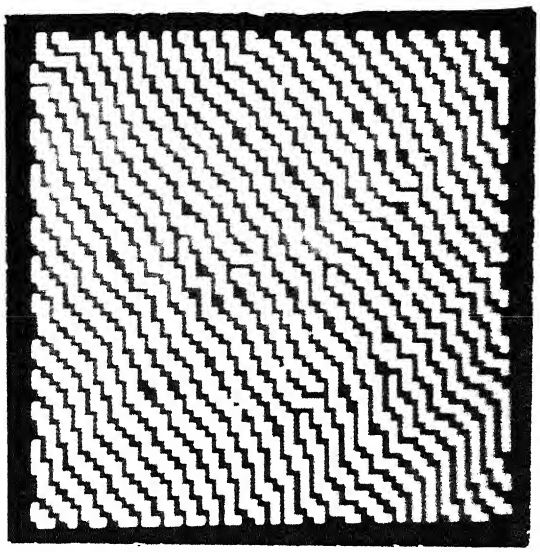


(a) idata

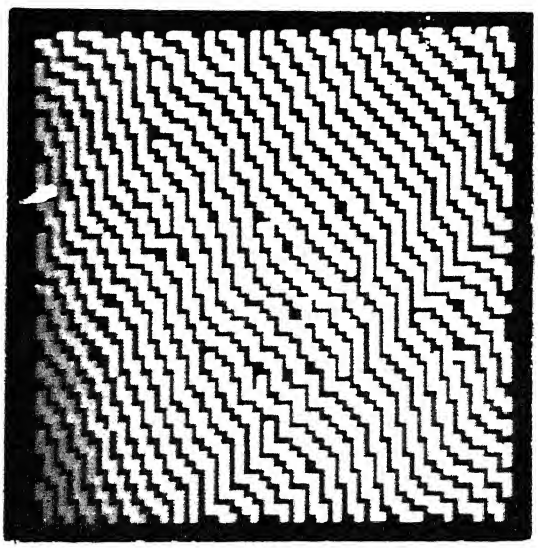


(b) idata1

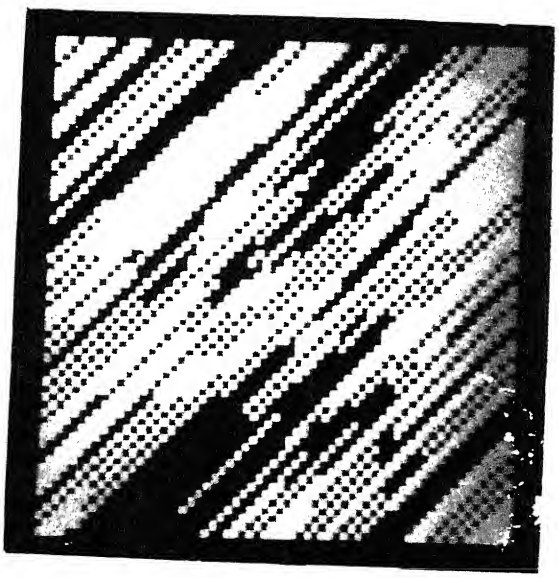
Figure 3.1: Starting Images



(a) starting from idata



(b) starting from idata1



(c) difference of (a) and (b)

Figure 3.2: Stabilised Images Without Annealing

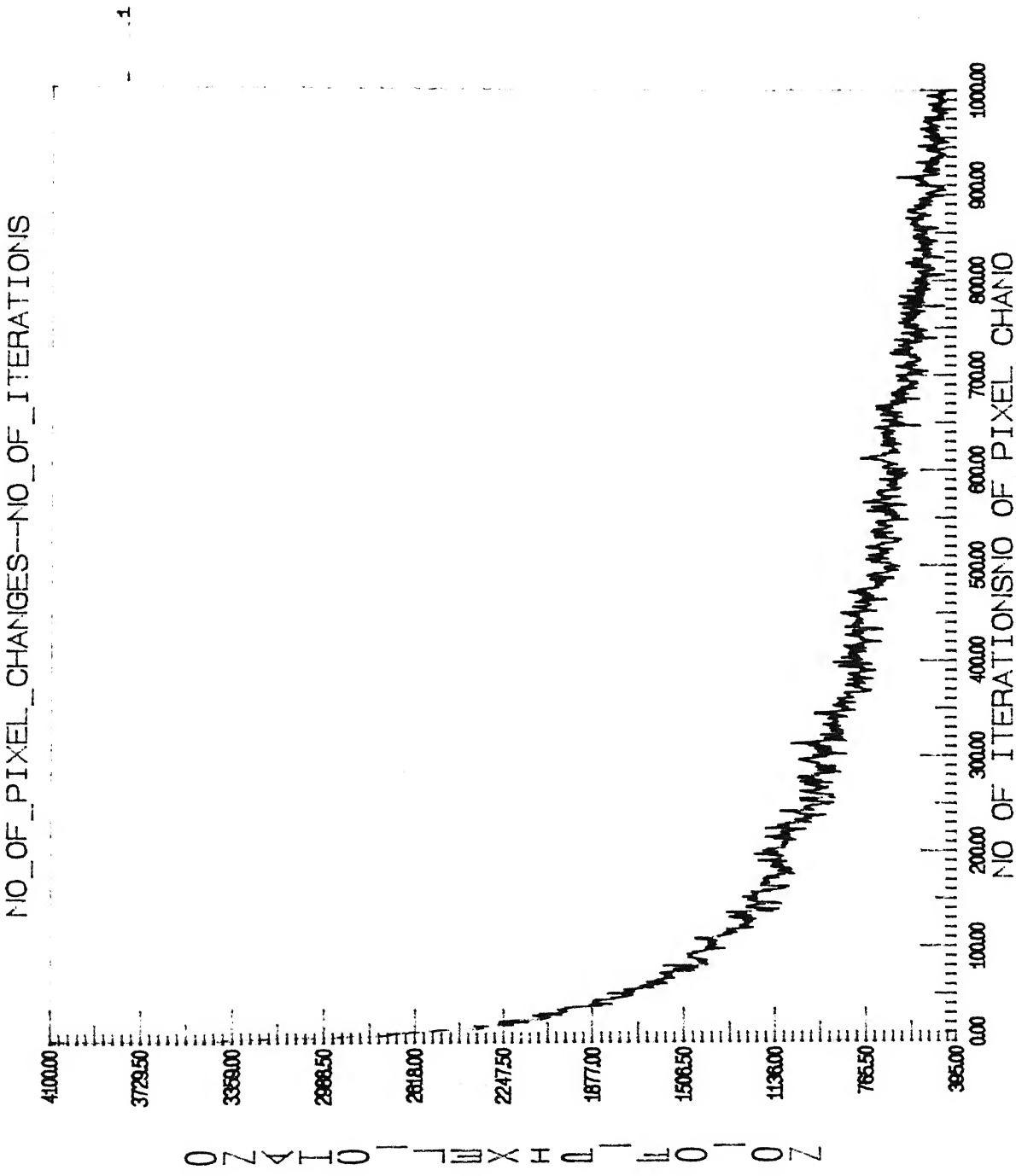
character of the distribution function. Thus, at lower T values, pixel values which have higher initial probabilities become much more probable as the iterations proceed. As the number of iterations increases beyond thousand, it is observed that the number of pixel changes in an iteration reduces to less than ten percent of the number of pixels. A graph is shown in Fig-3.3 that displays this behaviour. Fig-3.4 shows the stabilised images starting from idata and idata1 and the difference between the two. (See Appendix 3)

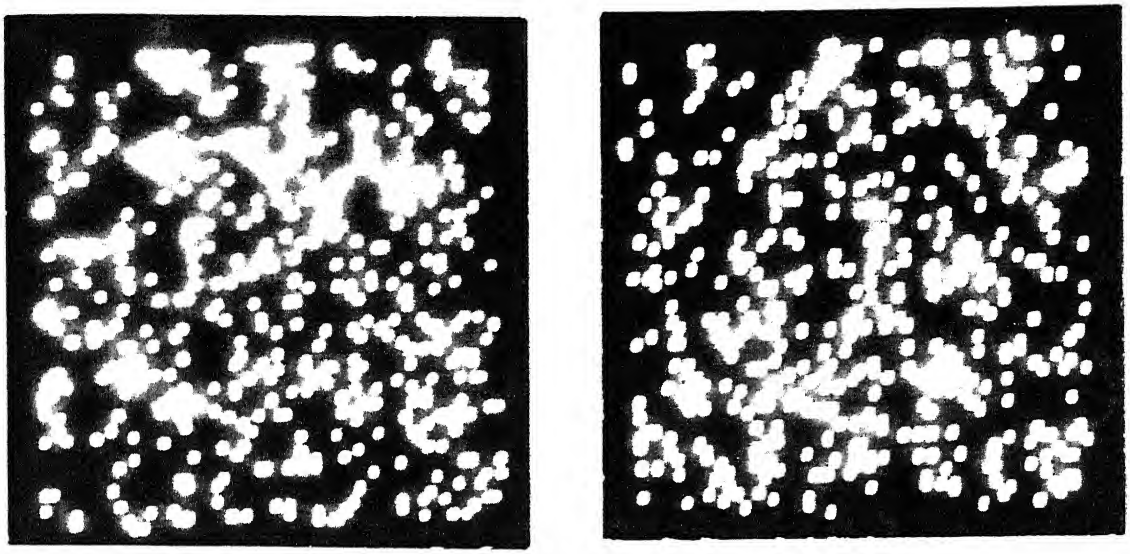
A variation of the algorithm for use with and without annealing is that steps 4–6 are changed. Now, irrespective of its present value, every pixel is updated to value -1 with probability $P(X[i,j]=-1)$, and to $+1$ with probability $P(X[i,j]=1)$. Thus, this altered procedure probabilistically updates the pixel values at each site without reference to the value at the site before the iteration. It has been observed that with this altered procedure, the images start stabilising only after a larger number of (500) iterations.

While this procedure undoubtedly ensures, as intended, an evolution of the starting image towards the desired probability distribution after a number of iterations, it has the disadvantage of being too slow.

3.4 SIGNIFICANCE OF THE VARIOUS MODEL PARAMETERS:

The parameter L affects the number of pixels of any grey level if all the other paranmeters are fixed. This is shown in the graph in Fig-3.5. If we increase L, the relative number of white pixels reduces. The parameters b_1 and b_3 are diagonal parameters. The parameter b_1 affects the 135° diagonal direction and b_3 affects

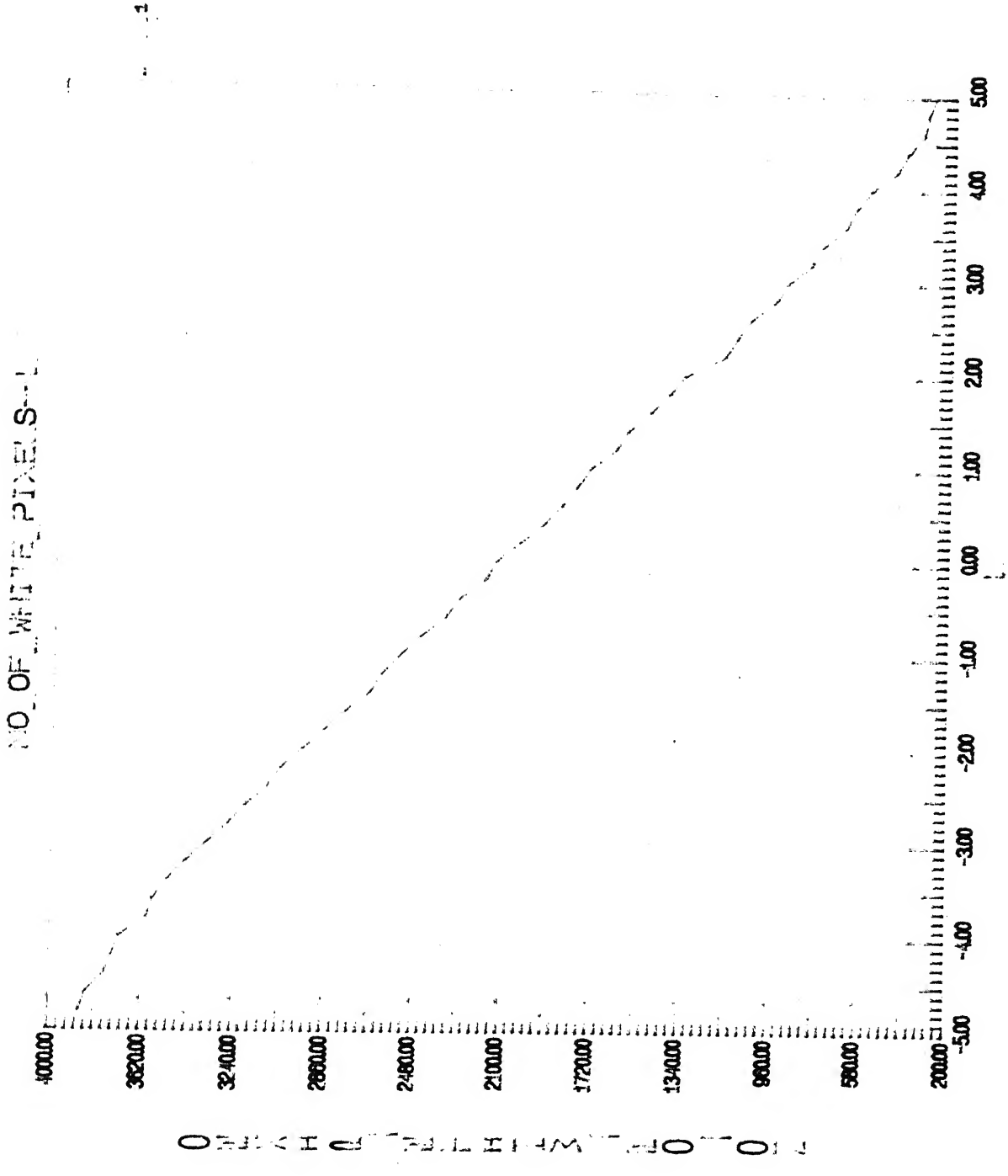

NO_OF_PIXEL_CHANGES--NO_OF_ITERATIONS
NO_OF_PIXEL_CHANGES
NO_OF_ITERATIONS
4100.00
3729.50
3359.00
2988.50
2618.00
2247.50
1877.00
1506.50
1136.00
765.50
395.00
0.00
100.00
200.00
300.00
400.00
500.00
600.00
700.00
800.00
900.00
1000.00



(a) starting from idata (b) starting from idata1

(c) difference of (a) and (b)

Figure 3.4: Stabilised Images With Annealing


NO_OF_WHITE_PIXELS
400000.00
362000.00
324000.00
286000.00
248000.00
210000.00
172000.00
134000.00
96000.00
58000.00
20000.00
-5.00
-4.00
-3.00
-2.00
-1.00
0.00
1.00
2.00
3.00
4.00
5.00

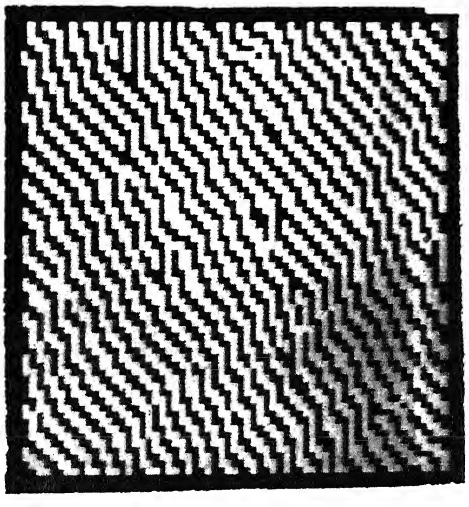
the 45° diagonal direction. This is illustrated in Fig-3.6(a) and Fig-3.6(c). The parameter b_1 dominates in Fig-3.6(a). The parameter b_3 dominates in Fig-3.6(c). The vertical direction is affected by parameter b_2. This is illustrated in Fig-3.6(b). Similarly the horizantal direction is affected by b_4 and this is illustrated in Fig-3.6(d). The parameters used for various images are summarised in Table-3.1.

TABLE 3.1: Parameters for Figure 3.6(a)–3.6(d)

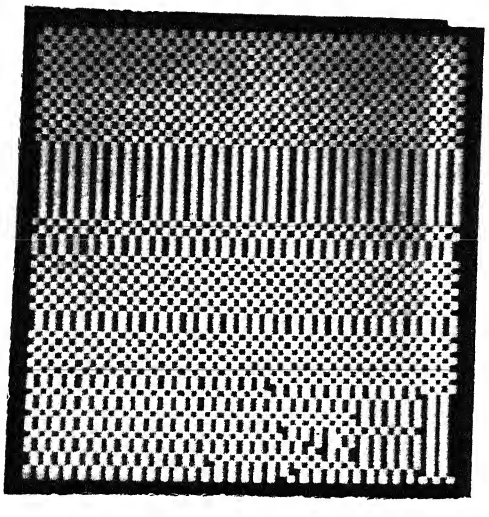
Images	Parameters				
	L	b_1	b_2	b_3	b_4
figure 3.6(a)	0.2	−65.0	0.1	0.1	0.1
figure 3.6(b)	0.2	0.1	−65.0	0.1	0.1
figure 3.6(c)	0.2	0.1	0.1	−65.0	0.1
figure 3.6(d)	0.2	0.1	0.1	0.1	−65.0

3.5 *PARAMETER ESTIMATION METHOD*

The parameter estimation method we present here is based on histogramming the image. The images considered are of binary type. If we take an η^2 neighbourhood system, there are eight neighbours for each pixel, and thus, 256 possible types of neighbourhoods. Let $x[i,j]$ be the image. In the image, the low level is represented as -1 and the high level is represented as $+1$. Let $y[i,j]$ represent the same image with the low level represented as 0 and the high level as $+1$. If value of the nearest top left diagonal pixel is taken as the least significant bit and the pixel above is taken as the next bit and so on, every pixel neighbourhood can be represented by an 8-bit binary number. The decimal value of the 8-bit binary

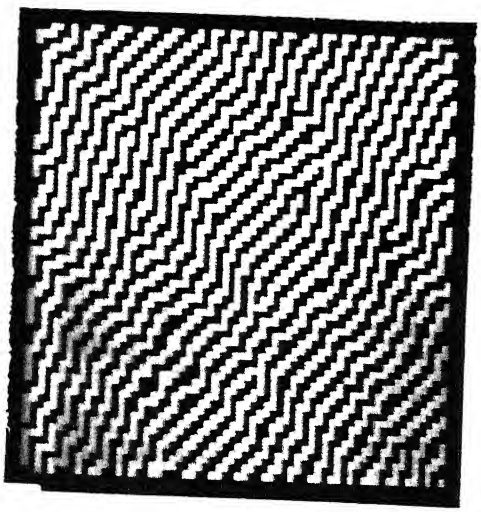


(a) left diagonal direction parameter dominant



(b) vertical direction parameter dominant

Figure 3.6: Left diagonal and vertical direction parameters dominant



(c) right diagonal direction parameter dominant

(d) horizontal direction parameter dominant

Figure 3.6: Right diagonal and horizontal direction parameters dominant

number representing the neighbourhood can assume values between 0 and 255. This number shall be called the *state* of the pixel.

The strategy adopted is that we start scanning the image in a raster fashion and search for each type (out of the 256 possible types) of neighbourhood in the $y[i,j]$ image matrix. When a pixel is found to have a given type of neighbourhood k, the pixel value is checked in the matrix $x[i,j]$ and if it is $+1$ then the counter $n[k,+1]$ that counts the number of occurences of pixels with value $+1$ and neighbourhood k, is incremented; else $n[k,-1]$ is incremented. After the scanning is completed over the entire image the result in $n[k,+1]$ is equal to number of pixels with value $+1$ and with neighbourhood k. Let $N[k] = n[k,+1] + n[k,-1]$. The ratio $n[k,+1]/N[k] = P[k,+1]$ represents the empirically determined probability of a pixel taking value $+1$ when the neighbourhood is k. Similarly, $n[k,-1]/N[k] = P[k,-1]$. Equipped with this information, our goal is to find the parameters of the GD governing the statistical properties of the image. In other words we have to solve a set of nonlinear equations to get the unknown parameters.

Solving the nonlinear equations is a tedious task, but fortunately, these equations can be transformed into a set of simultaneous linear equations. Then these equations can be solved by standard methods such as Gauss elimination. We use this method to find the parameters of simulated images.

3.6 PARAMETER ESTIMATION BY SOLVING SIMULTANEOUS EQUATIONS

To obtain a set of simultaneous equations from the nonlinear equations, we divide $P[k,+1]$ by $P[k,-1]$ if $P[k,-1]$ is a nonzero quantity and take the logarithm of the result. Cases where $P[k,-1]$ is zero have not been considered. This provides us with a maximum of 256 simultaneous equations in 5 unknowns. The equations are solved using the Gauss elimination method.

$$\begin{bmatrix} k_{0,1} & k_{0,2} & k_{0,3} & k_{0,4} & k_{0,5} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{255,1} & k_{255,2} & k_{255,3} & k_{255,4} & k_{255,5} \end{bmatrix} \cdot \begin{bmatrix} L \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} ln\frac{P[0,+1]}{P[0,-1]} \\ \vdots \\ ln\frac{P[255,+1]}{P[255,-1]} \end{bmatrix}$$

The coefficients $k_{i,1}$ to $k_{i,5}$ in each row of the coefficient matrix are the coefficients of the parameters in the expression $U[-1,k] - U[+1,k]$.

This method is used to estimate the parameters of various simulated images, and then, with the estimated parameters, the images are again generated. The initial and final images are compared and results found to be satisfactory. Before presenting the algorithm, we give the precise definition of the type of a neighbourhood.

DEFINITION 1: A pixel (i,j) is said to have a neighbourhood of type k if the state $s[i,j]=k$ where the state of the pixel is computed by the following expression.

$$s[i,j] = 2^0 \cdot y[i-1,j-1] + 2^1 \cdot y[i-1,j] + 2^2 \cdot y[i-1,j+1] + 2^3 \cdot y[i,j+1] + 2^4 \cdot y[i+1,j+1] + 2^5 \cdot y[i+1,j] + 2^6 \cdot y[i+1,j-1] + 2^7 \cdot y[i,j-1] \quad \ldots 3.3$$

where

$$y[i,j] = \frac{(x[i,j] + 1)}{2}; \qquad \ldots 3.4$$

The algorithm is as follows:

INPUT: A BINARY IMAGE FILE

OUTPUT: PARAMETERS OF THE IMAGE.

STEP-1: Form the matrix $y[i,j]$ from $x[i,j]$ using equation (3.4)

STEP-2: Form the matrix $s[i,j]$

STEP-3: $i=1, j=1$

STEP-4: $N[s[i,j]] = N[s[i,j]] + 1$

STEP-5: *If* $y[i,j]=1$ *then* $n[s[i,j],+1] = n[s[i,j],+1] + 1$

else $n[s[i,j],-1] = n[s[i,j],-1] + 1$

STEP-6: $j=j+1$,

STEP-7: *If* $j=N+1$ *then* $i=i+1$ and $j=1$

STEP-8: *If* $i=N+1$ *then goto* STEP-9 *else goto* STEP-4

STEP-9: *For* $k=0$ *to* 255 *find* $P[k,+1]=n[k,+1]/N[k]$

and $P[k,-1]=n[k,-1]/N[k]$

STEP-10: Form the kX1 matrix $log(P[k,+1]/P[k,-1])$

STEP-11: Form the coefficient matrix of the matrix equation by taking the coefficients of the parameters in the expression $U[-1,k] - U[+1,k]$.

STEP-12: Apply Gauss elimination to solve for the parameters.

STEP-13: Stop

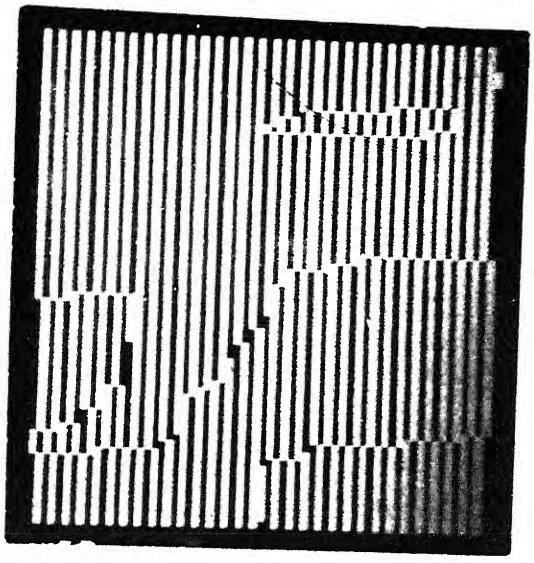
Regarding the algorithm the following points have to be kept in mind.

1. It has been assumed here that the parameters of the model are the same throughout the image. If this is not the case, the image has to be segmented into different regions depending upon the texture and histogramming has to be done seperately for each region. In that case, parameters of different regions would be different, and all the computations would have to be carried out separately for each of the regions.

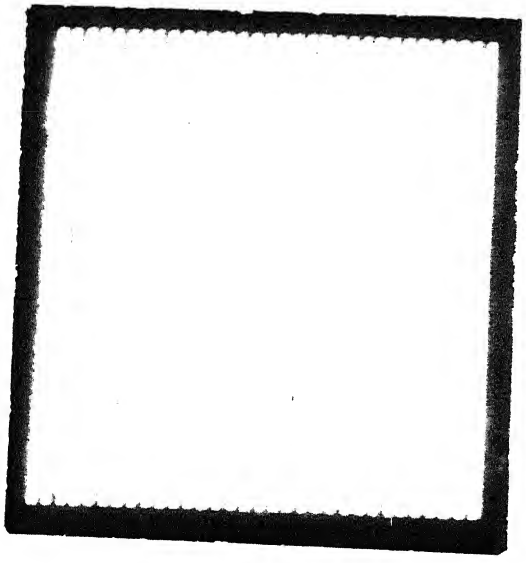
2. All the neighbourhoods of the pixels on the edges are not known: so the unknown pixels can be arbitrarily taken to be either $+1$ or -1. Alternatively, the outer pixels can be left out for the purpose of computing histograms about them and used only in the status of neighbours for the inner pixels while histogramming is done on the rest of the pixels.

3.7 *SIMULATION RESULTS*

The algorithm has been tested for the images Fig-3.7(a) to Fig-3.9(a). The parameters of the image 3.7(a) were estimated and from the estimated parameters the image 3.7(b) was generated. The texture of both the images are found to be the same, justifying the method. Similarly, images 3.8(a) and 3.9(a) were generated for different parameter sets and the parameters of the resulting were estimated. Finally, using the estimated parameters, the images 3.8(b) and 3.9(b) respectively

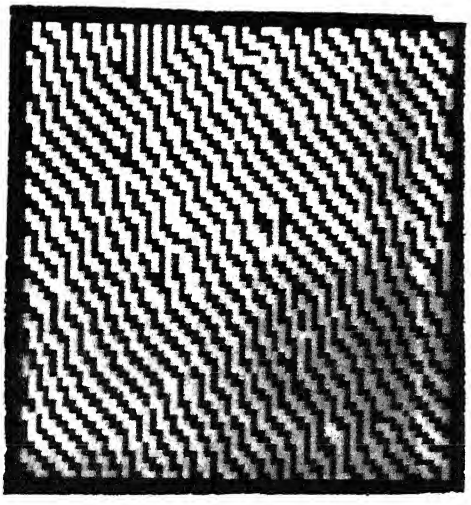


(a) original image

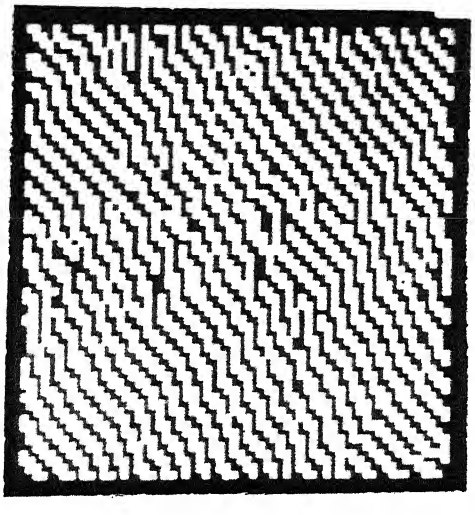


(b) image generated using estimated parameters

Figure 3.7: Original and Reconstructed Images

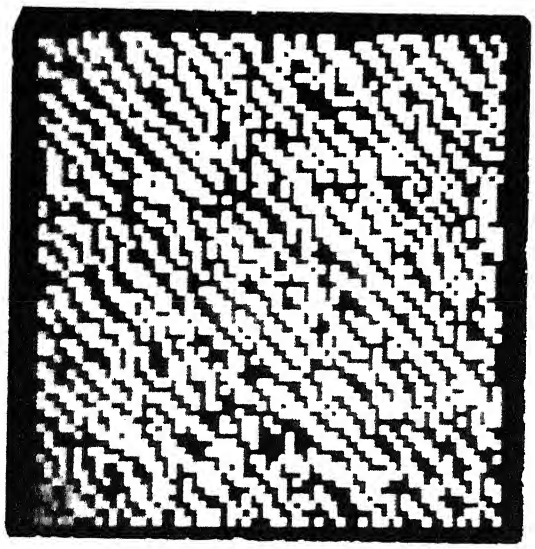


(a) original image

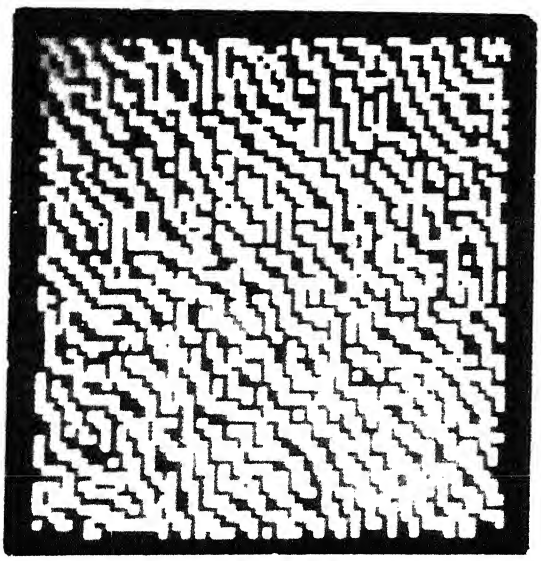


(b) image generated using estimated parameters

Figure 3.8: Original and Reconstructed Images



(a) original image



(b) image generated using estimated parameters

Figure 3.9. Original and Reconstructed Images

were generated. For the cases that were considered, good results have been obtained. The results of parameter estimation tests are summarized in Table 3.2.

TABLE 3.2: Original and Estimated Parameters

Images	Original Parameters					Estimated Parameters				
	L	b_1	b_2	b_3	b_4	L	b_1	b_2	b_3	b_4
figure 3.7	0.2	0.1	−0.6	0.1	0.1	0.2	0.22	−0.5	0.1	0.1
figure 3.8	0.2	−65.0	0.1	0.1	0.1	0.2	−60.0	0.1	0.1	0.09
figure 3.9	0.2	−0.6	−0.09	0.1	0.1	0.3	−0.4	−0.1	0.2	0.1

3.8 *PARALLEL IMPLEMENTATION OF THE ALGORITHM*

If the process of histogramming is done in parallel, there is a time saving. With a single processor, the time required is $O(N^2)$ if the image is an NXN array, whereas the parallel implementation suggested requires only O(log(N)) time. But this saving in time is obtained at the cost of an increase in the number of processors to $O(N^2)$.

The parallel machine proposed is shown in Fig-3.13. The process of searching for different types of neighbourhoods is made parallel. If all the eight neighbours of a pixel in the image $x[i,j]$ are combined to form the neighbourhood vector, then each element of the vector is either +1 or −1. The nearest left diagonal element is taken as the first element and the pixel above as the next element, and so on. Let us represent the vectors by V[i] where i ranges from 0 to 255.

A given type of neighbourhood is tested for a pixel by computing the dot product of the vector coressponding to each of the 256 types of neighbourhood vectors possible with the neighbourhood vector of the pixel. The value of the dot product with each neighbourhood is compared with 7 and if the product is greater than 7, it is decided that the pixel under study has the neighbourhood with whose vector, a dot product greater than 7 was obtained.

If the number of pixels is M, the number of pixels required is of $O(M)$. Once the neighbourhood of a pixel is found, the value of the pixel has to be tested. This can be done by multiplying the threshold output by the value of the centre pixel and if the value is $+1$, $n[k,+1]$ is incremented else $n[k,-1]$ is incremented. This procedure has to be repeated for all the pixels. Using the values of $n[k,+1]$ and $n[k,-1]$, the values of $P[k,+1]$ and $P[k,-1]$ have to be found for all k. Finally, the parameters can be evaluated using the Gauss elimination method.

Chapter 4

Applications of Gibbs Distribution Models

In the previous chapter, we presented a method of generating Gibbs distributed images. Effects of changing various model parameters on the image texture were also studied. The effect of different starting images on the texture of the stabilised image was also discussed. In this chapter, we are presenting two application of Gibbs distribution models to image compression. First, we show that the Gibbs distribution models can be applied for the purpose of image compression. Later, we apply the models for the purpose of texture classification.

4.1 IMAGE COMPRESSION METHOD

Using the same notation as in the previous chapters, we denote any image by the array $x[i,j]$, where i and j vary from 1 to N. From the results presented in Chapter3, it is amply evident that in applications where only the *texture* of an image has relevence, a knowledge of the parameters of the image suffices for reconstructing it. In practical cases, if we assume that 16 bits are required to

encode the information of each parameter, the compression ratio achieved is of the order of 50 for a 64X64 image and even greater if the image is larger.

But if the *actual* image has to be recovered, *i.e.*, more than just the texture information is desired, a very different approach is warranted. The possibility of reconstructing the whole image starting from a *part* of the image may be investigated. If the image is of binary type, then *NXN* bits of information are necessary to represent the image. Our purpose is to represent the same image in terms of a smaller number of bits, while at the same time, keeping the reconstruction error within bounds. The error is defined as the number of pixels on which the reconstructed and the original image differ measured as a percentage of the total number of pixels.

Such a means of image compression is possible only if values of some pixels of the image can be predicted in terms of the values of the other pixels. This is indeed possible in our model because the probability distribution of the value of a pixel is represented in terms of the values of its neighbours. So, if all the neighbours of any pixel are known as also the parameters of the image, the value of the pixel can be predicted. But by those means, the data compression obtained is only about 10 percent. To achieve further compression, we attempt at compromising upon the amount of data needed to record the values of all the neighbuors of a pixel. To do this by utilising the GD model, we find out which of the (four) parameters other than L is dominant (more negtive), and depending upon which is the most dominant, ensure that at least the neighbours corresponding to that parameter are known while predicting an unknown pixel. In practical implementations satisfying this criterion, there could be many ways of leaving out some of the pixels during compression.

STEP-6: $j=j+1$,

STEP-7: *If* $j<N+1$ *then goto* STEP-4.

STEP-8: $i=i+1$, $j=1$.

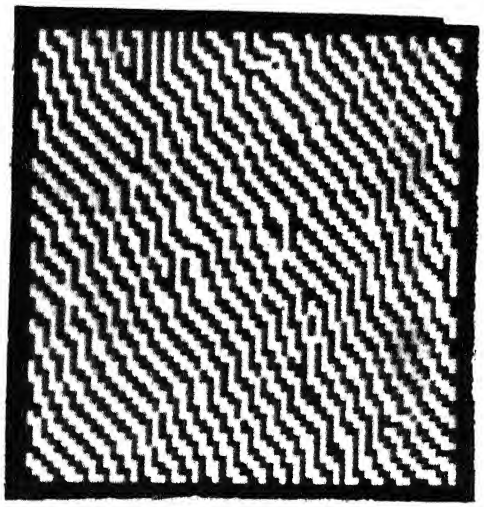
STEP-9: *If* $i<N+1$ *then goto* STEP-3.

STEP-10: *Stop*.

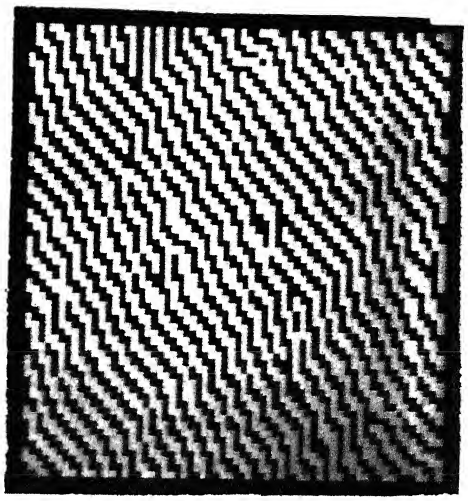
This algorithm uses the concept of maximum likelihood prediction. Use of a similar technique can be found in [31]. But there, as no modelling of the image is done, a table representing the probability distribution for each neighbourhood has to be maintained. Thus, in that case the compression ratio was drastically reduced due to that requirement. But in our case, only the parameters need to be transmitted along with the alternate rows. Of course, the reciever structure becomes more complex because the probabilities have to be evaluated.

This algorithm was implemented with different sets of parameters. The results obtained are summarised in Table-4.1. Fig-4.1(a) shows the original image and Fig-4.1(b) shows the reconstructed image and Fig-4.1(c) shows the difference between the original and reconstructed images. Other examples are given in figures 4.2 and 4.3.

Image	% Error
Figure 4.1	0.00
Figure 4.2	0.61
Figure 4.3	0.51



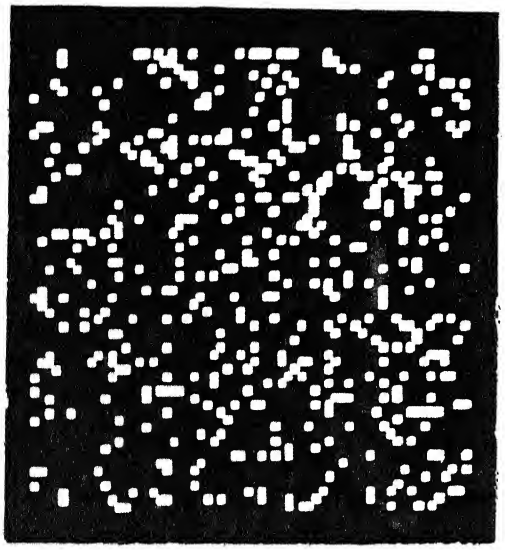
(a) original image



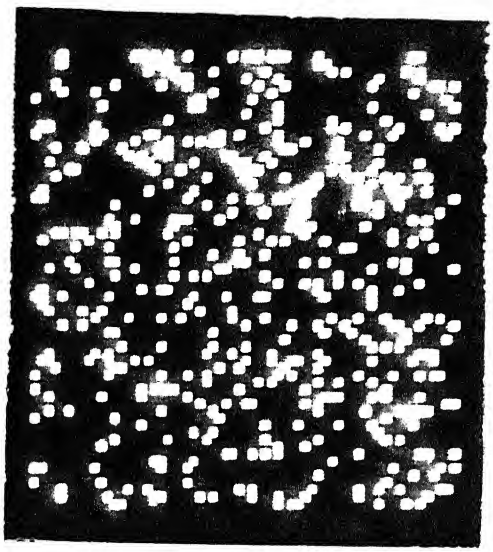
(b) recovered image

(c) error

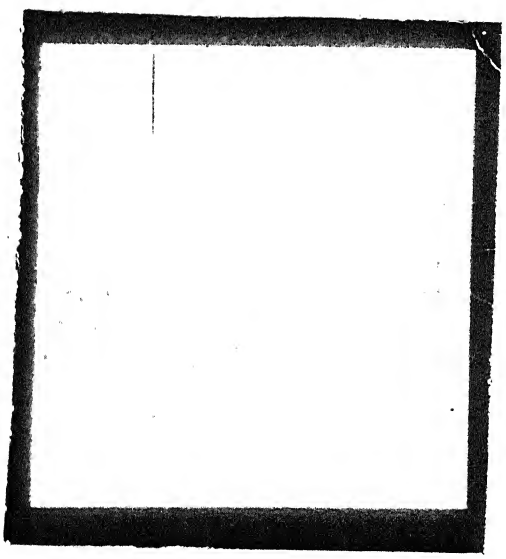
Figure 4.1: Original, Recovered and Error Images



(a) original image



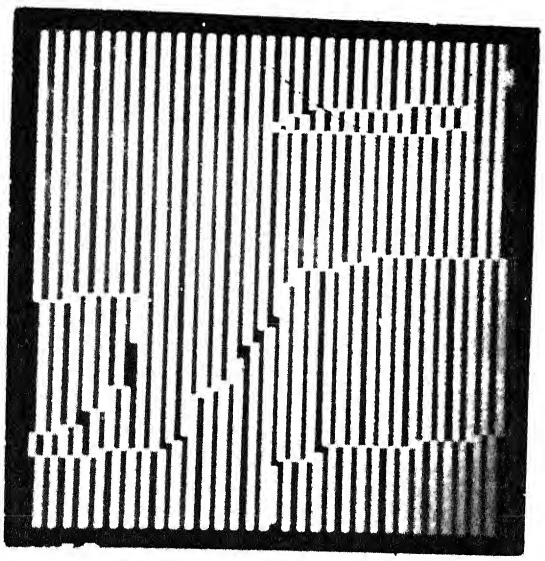
(b) recovered image



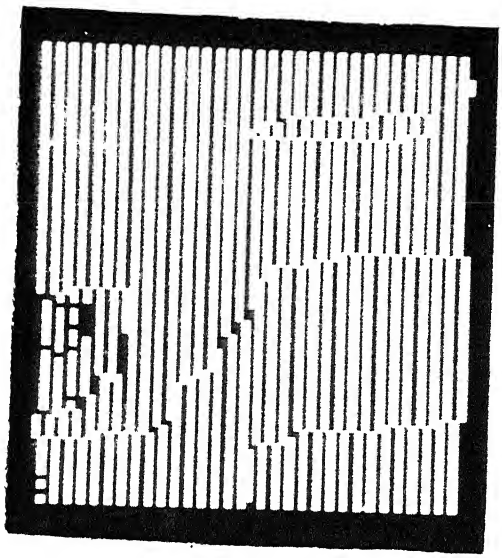
(c) error

CENTRAL LIBRARY
Acc. No. A113458

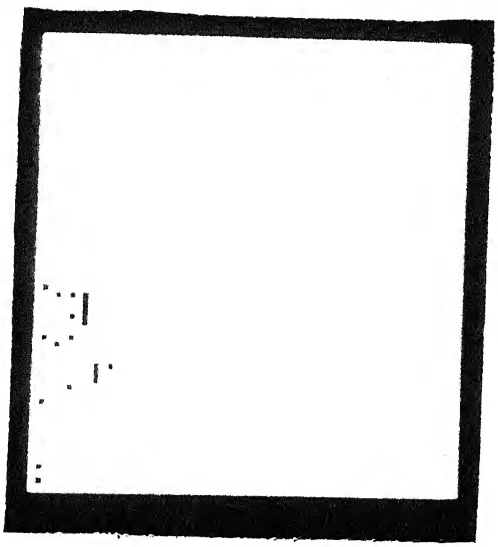
Figure 4.2: Original, Recovered and Error Images



(a) original image



(b) recovered image



(c) error

Figure 4.3: Original, Recovered and Error Images

4.2 TEXTURE CLASSIFICATION METHOD

The texture classification problem we intend to pose is as follows. Given N classes of images charecterised by their parameters (and therefore, their textures), and given a new image, how to classify the given image into one of the classes. In other words, it is a texture matching problem.

The strategy followed is that first, the parameters of the presented image are estimated. Then, an appropriate measure based on the difference of parameters of the presented image and the parameters characterising the various classes is evaluated The measure used is order equivalent to the Euclidean distance from the estimated parameter set of the image given and the parameter sets characterising the various classes. The distance from the estimate to any particular class is computed by calculating the sum of the squares of the differences between the corresponding parameter values. The prasented image is finally assigned to the class for which the measure is minimum. The algorithm used is as folllows.

INPUT: The image to be classified and parameters of the different classes.

OUTPUT: The class to which the presented image is assigned.

STEP-1: Find the parameters of the presented image.

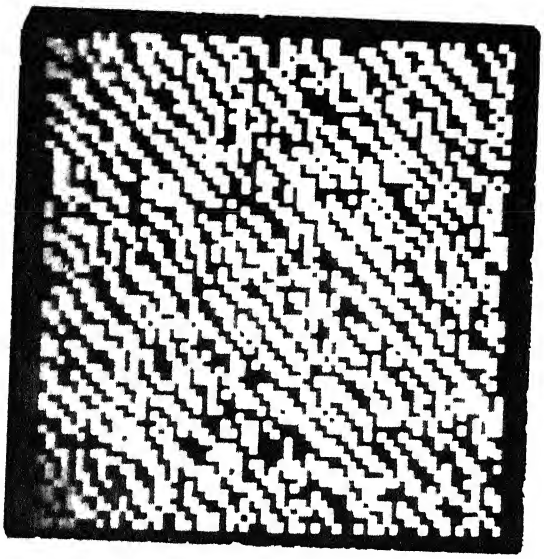
STEP-2: Evaluate $E(c_k) = \sum_i (b_i(c_k) - b_i)^2$ for each class c_k where $b_i(c_k)$ is the ith parameter of the kth class and b_i is the ith estimated parameter of the presented image.

STEP-3: Find k for which $E(c_k)$ is minimum. c_k is the class of the presented image.

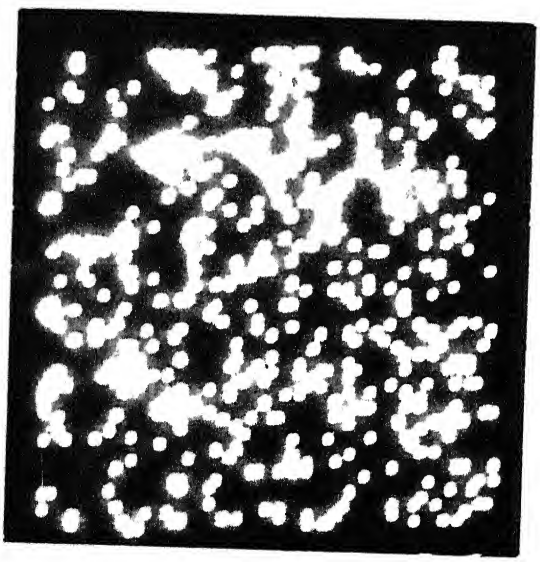
STEP-4: *Stop.*

The procedure can be made adaptive. In that case, as more and more images are classified, the parameter sets of the classes can be updated by averaging over all the parameter sets of the images allocated to the respective classas. If $E(c_k)$ for a given image is too large for every c_k, then the creation of a new class may be considered. If the least $E(c_k)$ does not point to a unique class on account of more than one class being at the same least distance, the classification procedure may be refined by assigning the image to that class among them for which the parameter dominant in the image is the closest.

For estimating the parameters, the parameter estimation method presented in Chapter 3 has been used. In Fig-4.4 the representative images of different classes are shown and in Fig-4.5 and 4.6, we show two presented images. The former image was assigned to c_2 and the latter to c_3.

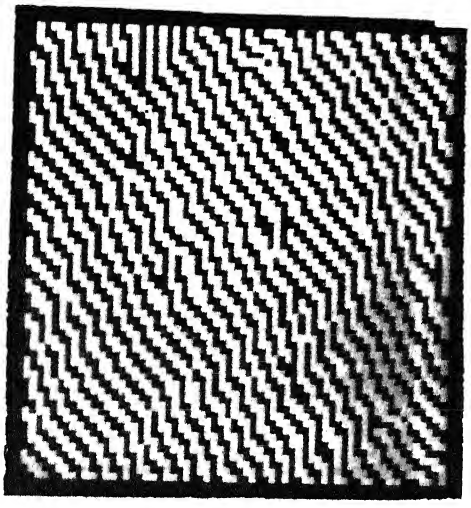


(a) class c_1

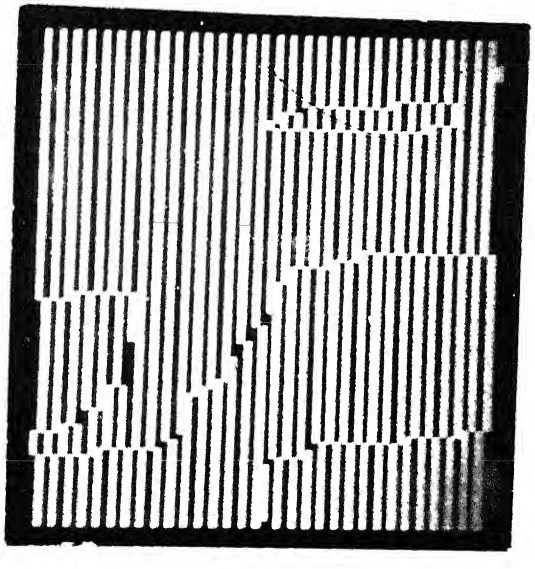


(b) class c_2

Figure 4.4



(c) class c_3



(d) class c_4

Figure 4.4

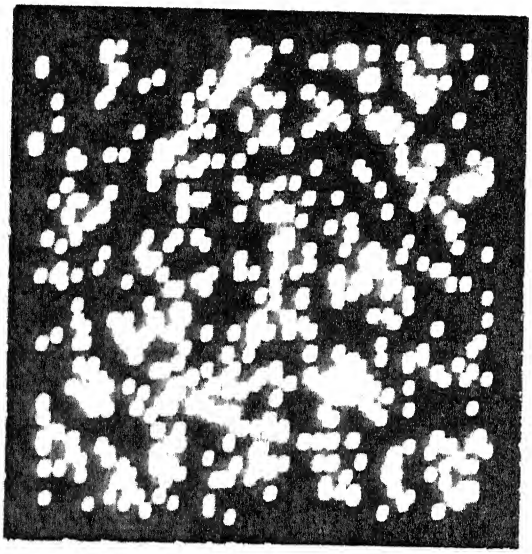


Figure 4.5: Presented Image Assigned to σ_2

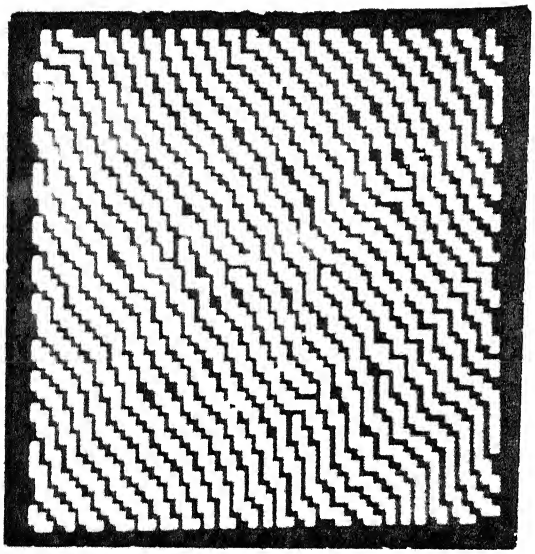


Figure 4.6: Presented Image Assigned to σ_3

Chapter 5

Conclusions and Scope for Future Work

5.1 *CONCLUSION*

In this thesis, we have studied some aspects of an image generation algorithm known as the spin-flip algorithm. We have observed that the final image texture is the same irrespective of the starting image. Effects of changing various parameters have also been studied.

A new parameter estimation algorithm is also proposed. In that algorithm, the parameters are assumed to be the same throughout the image. An η^2 type of neighbourhood system has been considered, so that there are a total of 256 types of neighbourhoods possible. Each type of neighbourhood is scanned in turn and its probability distribution at the center pixel noted. This information is used to form simultaneous equations with the parameters as the unknowns. This method was applied to various simulated images, and using the estimated parameters, images were again reconstructed. The algorithm was tested on various simulated images and it was found that the images generated using the estimated parameters had the same texture as the original image, thus justifying the algorithm.

Next, the Gibbs distribution model was applied to the image compression problem. Image compression was achieved by removing some pixels from images, and then recovering them by replacing them by their maximum likelihood estimates. On various simulated images a compression ratio of two was obtained without run-length coding with error levels less than 5%.

Finally, using the parameters of images, texture classification was attempted. The algorithm used was non-adaptive. The problem was posed as follows: given the parameters for various classes, and an image, how is the image to be classified?. A measure that depended upon the difference of parameters was evaluated between the estimated parameters of the given image and the parameters characterising the various classes, and the image was attached to the class with which the measure was found to be minimum. The method was tried with various input textures and in all cases taken up, correct classification was obtained.

5.2 *SUGGESTIONS FOR FUTURE WORK*

In the case of image compression, there are many aspects which have not been investigated because of lack of time. An analytical relation between the number of pixels removed and the error in construction is required. Also, the locations of pixels removed while effecting image compression also affect the error; this too has to be investigated more rigorously.

The use of the image compression method on real images was not tried out. For real images, regions of similar texture have to first be identified, and for

each region, parameters have to be estimated independently. Moreover, the parameters for different regions have all to be transmitted.

Finally, the texture classification method can be made adaptive. The image classification exercise is started with some initially chosen sets of parameters for each class. As more and more images are classified with the use of these original parameter sets, it is possible to refine the classification with the help of the knowledge gained from the images already classified. One method of making the classifier adaptive is the following: every time a new image is classified, the parameters of the class to which it belongs should be appropriately updated. By properly selecting the form of this updating procedure, it may be ensured that while newer images get classified as per the updated parameter sets representing each class, the classification of no previous image will need to be revised on account of the continual updating of the parameter sets.

References

1. R. L. Dobrushin, The description of a random field by means of conditional probabilities and conditions of its regularity, Theory Prob. Appli. Vol-13, pp. 197-224, 1968.

2. E. Wong, Recursive causal linear filtering for two dimensional random fields, IEEE Trans. Infor. Theory, Vol. IT-24, PP. 50-59, January 1978.

3. J. W. Woods, Two dimensional discrete Markovian fields, IEEE Trans. Infor. Theory, Vol IT-18, PP. 232-240, March 1972.

4. K. Abend, T. J. Hartely, and L. N. Kanal, Classification of binary random patterns, IEEE Trans. Infor. Theory, Vol. IT-11, PP. 538-554 October 1965.

5. J. W. Woods, and C. H. Radewan, Kalman filtering in two dimensions, IEEE Trans. Infor. Theory, Vol. IT-23, PP. 473-482, July 1977.

6. R. Chellappa, and R. L. Kashyap, Digital image restoration using interaction models, IEEE Trans. ASSP Vol. ASSP-30, PP. 461-472, June 1982.

7. H Kaufman, J W Woods, V K. Ingle, R. Mediquilla and A. Radpour, Recursive Estimation a multiple model approach, Proc. 18th Conf. Decision. Cont., Ft. Lauderdale, Fl Dec 1979.

8 C. W Therrien, Linear filtering models for texture classification and segmentation. Proc 5th Int. Conf. Pattern Recognition, Miami, FL, Dec.1980.

9 F R Hanson and H Elliott, Image segmentation using simple Markov field Models, Comp Graph Im Proc., Vol. 20, PP. 101-132, 1982.

10 E Ising Zeitschrift Physik, Vol 31,P.253, 1925.

11 D Ruelle. Statistical Mechanics, Benjamin, New York, 1977.

12 P A Flinn, Monte Carlo calculation of phase separation in a two dimensional Ising system, Journ Statist Phys. Vol. 10, PP. 89-97, 1974.

13 J Besag, Spatial interaction and statistical analysis of lattice systems, Journ Roy. Statistical Soc Series, Vol. 36, PP. 192-226, 1974.

14 M B Averintzev, On a method of describing discrete parameters of random fields, Problemy Peredecci Informacci Vol.6 PP. 100-109,1970.

15 F Spitzer, Markov Random Fields and Gibbs Ensembles, Ameri. Math. Mo., Vol. 78, PP 142-154, February 1971.

16 R Kinderman and J L Snell, Markov random fields and their applications, Providence, R.I. American Mathematical Society, 1980.

17 M Hassner and J Skalansky, The use of Markov random fields as models of texture Compt. Grap. Im. Proc. Vol. 12, PP. 357-370. 1980.

18. G. R. Cross and A. K. Jain, Markov random field texture models, IEEE Tran. Pat. Analy. Mach. Intel. Vol. PAMI-5, PP-25-39, January, 1983.

19. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller Equation of state calculations by fast computing machines, Journ. Chem. Phys., Vol. 21, PP, 1087-1092, June, 1953.

20. D. Geman and S. Geman, Stochastic relaxation, Gibbs distribution, and the Baysian restoration of images, IEEE Trans. Pattern Anal. Machine intell., Vol. PAMI-6, No-6, PP721-741, November, 84.

21. F. S. Cohen and D. B. Copper, Real time textured image segmentation based on non-causal Markovian random field models, Proc. of SPIE Conf. Intel. Robot. Cambridge, MA., November 1983.

22. H. Ellott, H. Derin, R. Cristi and D. Geman, Applications of Gibbs distributions to image segmentation, presented at the IEEE Int. Conf. ASSP, San Diego CA, Mar. 1984.

23. S. Lakshmanan and H. Derin, Simultaneous parameter estimation and segmentation of Gibbs random fiels using simulated annealing. IEEE Trans. Pat. Analy. Mach. Intel. Vol. 11, No-8, PP. 799-83, August 1989.

24. D. Geiger and F. Girosi, Parallel and deterministic algorithms for MRFs: Surface reconstruction. Vol. 13, No-5, IEEE Trans Pat. Analy. and Mach. Intel., PP. 401-412, May 1991.

25. L. Onural, Generating connected textured fractal pattern using Markov random fields, IEEE Trans Pat. Analy. and Mach. Intel., PP. 819-825, Vol. 13, No-8, August, 1991.

26. A. Veijanen, A simulation-based estimator for hidden Markov random fields, IEEE Trans Pat. Analy. and Mach. Intel., PP. 825-830, August, 1991.

27. F. S. Cohen, Z. Fan and M. A. Patel, Classification of rotated and scaled textured images using Gaussian Markov random field models, IEEE Trans Pat. Analy. and Mach. Intel., PP. 192-202, Vol. 13, No-8, February, 1991.

28. P. A. P. Moran, A Gaussian Markovian process on a square lattice, J. Appli. Prob. Vol. 10, PP 54-62, 1973.

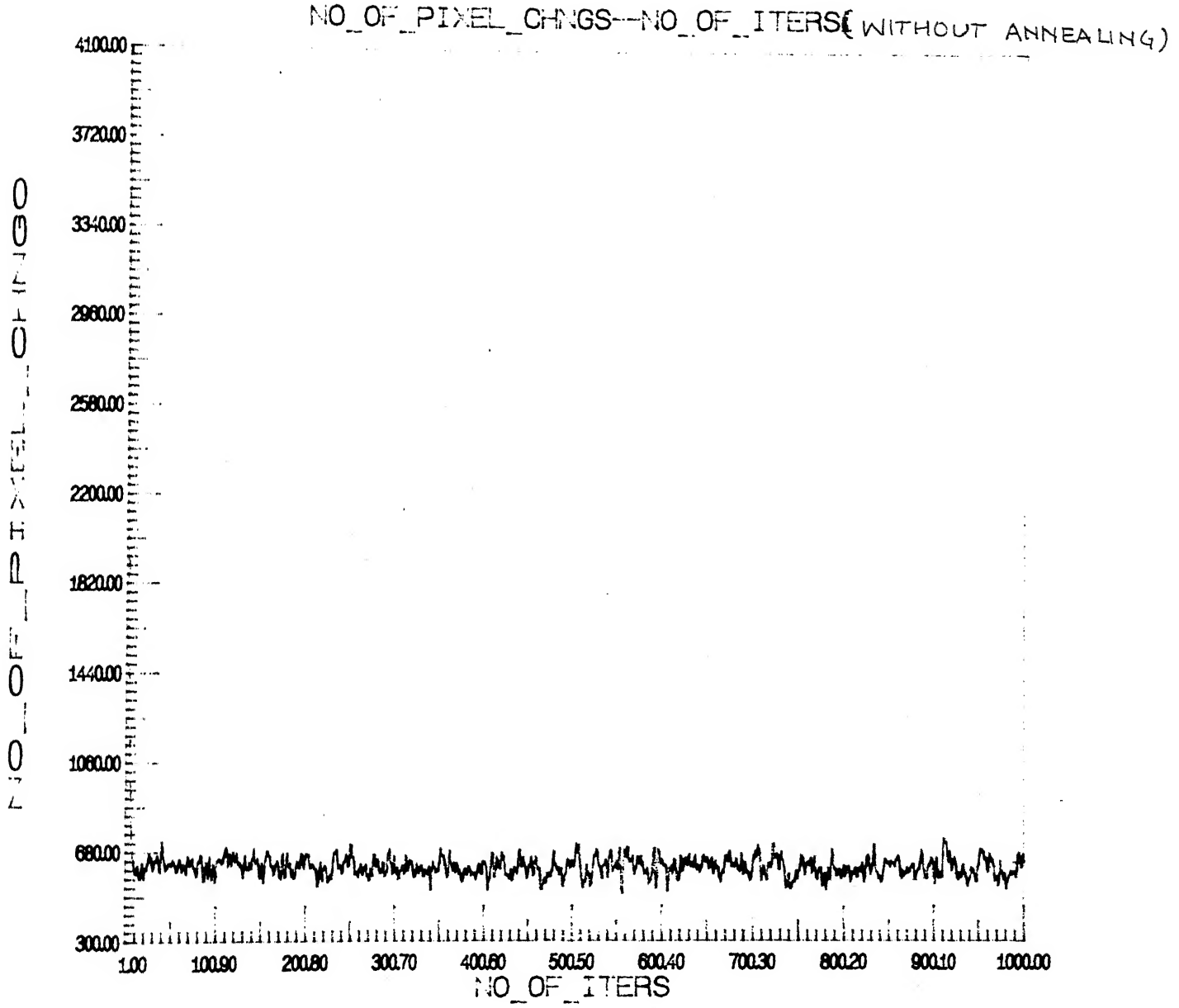
29. R. Chellappa and S. Chatterjee, Classification of textures using Gaussian Markov random fields, IEEE Trans. Acoust. Speech Sig. Proc., August, 1985.

30. H. Derin, The use of Gibbs distributions in image processing, comm. and network, A survey of recent advances, Springer-Verlag, 1986, PP. 266-298.

31. M. E. Kanefsky, Image compression via self-error correcting two-dimensional coding, comm. and network, A survey of recent advances, Springer-Verlag, 1986, PP. 266-298.

32. F. S. Cohen, Z. Fan, and S. Attali, Automated inspection of textile fabrics using textural models, IEEE Trans Pat. Analy. and Mach. Intel., PP. 803-808, Vol 13, No-8, August, 1991.

APPENDIX-1


NO_OF_PIXEL_CHNGS--NO_OF_ITERS(WITHOUT ANNEALING)
NO_OF_PIXEL_CHNGS
4100.00
3720.00
3340.00
2960.00
2580.00
2200.00
1820.00
1440.00
1060.00
680.00
300.00
1.00
100.90
200.80
300.70
400.60
500.50
600.40
700.30
800.20
900.10
1000.00
NO_OF_ITERS

3	1.00000E+00	1.00000E+00	0.00000E+00
4	1.00000E+00	1.00000E+00	0.00000E+00
5	1.00000E+00	1.00000E+00	0.00000E+00
6	1.00000E+00	1.00000E+00	0.00000E+00
7	1.00000E+00	1.00000E+00	0.00000E+00
8	1.00000E+00	1.00000E+00	0.00000E+00
9	1.00000E+00	1.00000E+00	0.00000E+00
10	1.00000E+00	1.00000E+00	0.00000E+00
11	1.00000E+00	1.00000E+00	0.00000E+00
12	1.00000E+00	1.00000E+00	0.00000E+00
13	1.00000E+00	1.00000E+00	0.00000E+00
14	1.00000E+00	1.00000E+00	0.00000E+00
15	1.00000E+00	1.00000E+00	0.00000E+00
16	1.00000E+00	1.00000E+00	0.00000E+00
17	1.00000E+00	1.00000E+00	0.00000E+00
18	1.00000E+00	1.00000E+00	0.00000E+00
19	1.00000E+00	1.00000E+00	0.00000E+00
20	1.00000E+00	1.00000E+00	0.00000E+00
21	1.00000E+00	1.00000E+00	0.00000E+00
22	1.00000E+00	1.00000E+00	0.00000E+00
23	1.00000E+00	1.00000E+00	0.00000E+00
24	1.00000E+00	1.00000E+00	0.00000E+00
25	1.00000E+00	1.00000E+00	0.00000E+00
26	1.00000E+00	1.00000E+00	0.00000E+00
27	1.00000E+00	1.00000E+00	0.00000E+00
28	1.00000E+00	1.00000E+00	0.00000E+00
29	1.00000E+00	1.00000E+00	0.00000E+00
30	1.00000E+00	1.00000E+00	0.00000E+00
31	1.00000E+00	1.00000E+00	0.00000E+00
32	1.00000E+00	1.00000E+00	0.00000E+00
33	1.00000E+00	1.00000E+00	0.00000E+00
34	1.00000E+00	1.00000E+00	0.00000E+00
35	1.00000E+00	1.00000E+00	0.00000E+00
36	1.00000E+00	1.00000E+00	0.00000E+00
37	1.00000E+00	1.00000E+00	0.00000E+00
38	1.00000E+00	1.00000E+00	0.00000E+00
39	1.00000E+00	1.00000E+00	0.00000E+00
40	1.00000E+00	1.00000E+00	0.00000E+00
41	1.00000E+00	1.00000E+00	0.00000E+00
42	1.00000E+00	1.00000E+00	0.00000E+00
43	1.00000E+00	1.00000E+00	0.00000E+00
44	1.00000E+00	1.00000E+00	0.00000E+00
45	1.00000E+00	1.00000E+00	0.00000E+00
46	1.00000E+00	1.00000E+00	0.00000E+00
47	1.00000E+00	1.00000E+00	0.00000E+00
48	1.00000E+00	1.00000E+00	0.00000E+00
49	1.00000E+00	1.00000E+00	0.00000E+00
50	1.00000E+00	1.00000E+00	0.00000E+00
51	1.00000E+00	1.00000E+00	0.00000E+00
52	1.00000E+00	1.00000E+00	0.00000E+00
53	1.00000E+00	1.00000E+00	0.00000E+00
54	1.00000E+00	1.00000E+00	0.00000E+00
55	1.00000E+00	1.00000E+00	0.00000E+00
56	7.14286E-01	1.00000E+00	-2.85714E-01
57	1.00000E+00	1.00000E+00	0.00000E+00
58	1.00000E+00	1.00000E+00	0.00000E+00
59	1.00000E+00	1.00000E+00	0.00000E+00
60	5.00000E-01	1.00000E+00	-5.00000E-01
61	1.00000E+00	1.00000E+00	0.00000E+00
62	1.00000E+00	1.00000E+00	0.00000E+00

135	4.33333E-01	4.28571E-01	4.76202E-03
136	1.00000E+00	1.00000E+00	0.00000E+00
137	1.00000E+00	1.00000E+00	0.00000E+00
138	1.00000E+00	1.00000E+00	0.00000E+00
139	1.00000E+00	1.00000E+00	0.00000E+00
140	1.00000E+00	1.00000E+00	0.00000E+00
141	1.00000E+00	1.00000E+00	0.00000E+00
142	1.00000E+00	1.00000E+00	0.00000E+00
143	1.00000E+00	1.00000E+00	0.00000E+00
144	1.00000E+00	1.00000E+00	0.00000E+00
145	1.00000E+00	1.00000E+00	0.00000E+00
146	1.00000E+00	1.00000E+00	0.00000E+00
147	1.00000E+00	1.00000E+00	0.00000E+00
148	1.00000E+00	1.00000E+00	0.00000E+00
149	1.00000E+00	1.00000E+00	0.00000E+00
150	1.00000E+00	1.00000E+00	0.00000E+00
151	1.00000E+00	1.00000E+00	0.00000E+00
152	1.00000E+00	1.00000E+00	0.00000E+00
153	1.00000E+00	1.00000E+00	0.00000E+00
154	1.00000E+00	1.00000E+00	0.00000E+00
155	1.00000E+00	1.00000E+00	0.00000E+00
156	1.00000E+00	1.00000E+00	0.00000E+00
157	1.00000E+00	1.00000E+00	0.00000E+00
158	1.00000E+00	1.00000E+00	0.00000E+00
159	1.00000E+00	1.00000E+00	0.00000E+00
160	1.00000E+00	1.00000E+00	0.00000E+00
161	1.00000E+00	1.00000E+00	0.00000E+00
162	1.00000E+00	1.00000E+00	0.00000E+00
163	1.00000E+00	1.00000E+00	0.00000E+00
164	1.00000E+00	1.00000E+00	0.00000E+00
165	1.00000E+00	1.00000E+00	0.00000E+00
166	1.00000E+00	1.00000E+00	0.00000E+00
167	1.00000E+00	1.00000E+00	0.00000E+00
168	1.00000E+00	1.00000E+00	0.00000E+00
169	1.00000E+00	1.00000E+00	0.00000E+00
170	1.00000E+00	1.00000E+00	0.00000E+00
171	1.00000E+00	1.00000E+00	0.00000E+00
172	1.00000E+00	1.00000E+00	0.00000E+00
173	1.00000E+00	1.00000E+00	0.00000E+00
174	1.00000E+00	1.00000E+00	0.00000E+00
175	1.00000E+00	1.00000E+00	0.00000E+00
176	1.00000E+00	1.00000E+00	0.00000E+00
177	1.00000E+00	1.00000E+00	0.00000E+00
178	1.00000E+00	1.00000E+00	0.00000E+00
179	1.00000E+00	1.00000E+00	0.00000E+00
180	1.00000E+00	1.00000E+00	0.00000E+00
181	1.00000E+00	1.00000E+00	0.00000E+00
182	1.00000E+00	1.00000E+00	0.00000E+00
183	1.00000E+00	1.00000E+00	0.00000E+00
184	1.00000E+00	1.00000E+00	0.00000E+00
185	1.00000E+00	1.00000E+00	0.00000E+00
186	1.00000E+00	1.00000E+00	0.00000E+00
187	1.00000E+00	1.00000E+00	0.00000E+00
188	1.00000E+00	1.00000E+00	0.00000E+00
189	1.00000E+00	1.00000E+00	0.00000E+00
190	1.00000E+00	1.00000E+00	0.00000E+00
191	1.00000E+00	1.00000E+00	0.00000E+00
192	1.00000E+00	1.00000E+00	0.00000E+00
193	1.00000E+00	1.00000E+00	0.00000E+00
194	1.00000E+00	1.00000E+00	0.00000E+00

197	1.00000E+00	1.00000E+00	0.00000E+00
198	1.00000E+00	1.00000E+00	0.00000E+00
199	1.11111E-01	5.00000E-02	6.11110E-02
200	1.00000E+00	1.00000E+00	0.00000E+00
201	1.00000E+00	1.00000E+00	0.00000E+00
202	1.00000E+00	1.00000E+00	0.00000E+00
203	1.00000E+00	1.00000E+00	0.00000E+00
204	1.00000E+00	1.00000E+00	0.00000E+00
205	1.00000E+00	1.00000E+00	0.00000E+00
206	1.00000E+00	1.00000E+00	0.00000E+00
207	1.00000E+00	1.00000E+00	0.00000E+00
208	1.00000E+00	1.00000E+00	0.00000E+00
209	1.00000E+00	1.00000E+00	0.00000E+00
210	1.00000E+00	1.00000E+00	0.00000E+00
211	1.00000E+00	0.00000E+00	1.00000E+00
212	1.00000E+00	1.00000E+00	0.00000E+00
213	1.00000E+00	1.00000E+00	0.00000E+00
214	1.00000E+00	1.00000E+00	0.00000E+00
215	1.00000E+00	1.00000E+00	0.00000E+00
216	1.00000E+00	1.00000E+00	0.00000E+00
217	1.00000E+00	1.00000E+00	0.00000E+00
218	1.00000E+00	1.00000E+00	0.00000E+00
219	1.00000E+00	1.00000E+00	0.00000E+00
220	1.00000E+00	1.00000E+00	0.00000E+00
221	1.00000E+00	1.00000E+00	0.00000E+00
222	1.00000E+00	1.00000E+00	0.00000E+00
223	1.00000E+00	1.00000E+00	0.00000E+00
224	1.00000E+00	1.00000E+00	0.00000E+00
225	1.00000E+00	1.00000E+00	0.00000E+00
226	1.00000E+00	1.00000E+00	0.00000E+00
227	1.00000E+00	1.00000E+00	0.00000E+00
228	1.00000E+00	1.00000E+00	0.00000E+00
229	1.00000E+00	1.00000E+00	0.00000E+00
230	1.00000E+00	1.00000E+00	0.00000E+00
231	1.00000E+00	1.00000E+00	0.00000E+00
232	1.00000E+00	1.00000E+00	0.00000E+00
233	1.00000E+00	1.00000E+00	0.00000E+00
234	1.00000E+00	1.00000E+00	0.00000E+00
235	1.00000E+00	1.00000E+00	0.00000E+00
236	1.00000E+00	1.00000E+00	0.00000E+00
237	1.00000E+00	1.00000E+00	0.00000E+00
238	1.00000E+00	1.00000E+00	0.00000E+00
239	1.00000E+00	1.00000E+00	0.00000E+00
240	1.00000E+00	1.00000E+00	0.00000E+00
241	1.00000E+00	1.00000E+00	0.00000E+00
242	1.00000E+00	1.00000E+00	0.00000E+00
243	1.00000E+00	1.00000E+00	0.00000E+00
244	1.00000E+00	1.00000E+00	0.00000E+00
245	1.00000E+00	1.00000E+00	0.00000E+00
246	1.00000E+00	1.00000E+00	0.00000E+00
247	1.00000E+00	1.00000E+00	0.00000E+00
248	1.00000E+00	1.00000E+00	0.00000E+00
249	1.00000E+00	1.00000E+00	0.00000E+00
250	1.00000E+00	1.00000E+00	0.00000E+00
251	1.00000E+00	9.33333E-01	6.66670E-02
252	1.00000E+00	1.00000E+00	0.00000E+00
253	1.00000E+00	1.00000E+00	0.00000E+00
254	1.00000E+00	1.00000E+00	0.00000E+00
255	1.00000E+00	1.00000E+00	0.00000E+00

3	8.84615E-01	8.84615E-01	8.20520E-02
4	8.98305E-01	8.50515E-01	4.77900E-02
5	9.09091E-01	8.26087E-01	8.30040E-02
6	8.96552E-01	9.62963E-01	-6.64110E-02
7	9.18919E-01	8.62069E-01	5.68500E-02
8	8.75706E-01	9.09091E-01	-3.33850E-02
9	9.23077E-01	8.82353E-01	4.07240E-02
10	8.40000E-01	8.96552E-01	-5.65521E-02
11	6.66667E-01	5.00000E-01	1.66667E-01
12	8.62069E-01	9.13043E-01	-5.09740E-02
13	1.00000E+00	6.00000E-01	4.00000E-01
14	1.00000E+00	1.00000E+00	0.00000E+00
15	1.00000E+00	6.66667E-01	3.33333E-01
16	8.75000E-01	8.48958E-01	2.60420E-02
17	9.41176E-01	7.74194E-01	1.66982E-01
18	1.00000E+00	7.66667E-01	2.33333E-01
19	7.50000E-01	7.50000E-01	0.00000E+00
20	9.04762E-01	9.62963E-01	-5.82010E-02
21	1.00000E+00	1.00000E+00	0.00000E+00
22	1.00000E+00	1.00000E+00	0.00000E+00
23	6.66667E-01	7.50000E-01	-8.33330E-02
24	1.00000E+00	8.00000E-01	2.00000E-01
25	5.00000E-01	8.00000E-01	-3.00000E-01
26	7.50000E-01	1.00000E+00	-2.50000E-01
27	1.00000E+00	1.00000E+00	0.00000E+00
28	9.79592E-01	9.04762E-01	7.48300E-02
29	1.00000E+00	7.50000E-01	2.50000E-01
30	1.00000E+00	1.00000E+00	0.00000E+00
31	6.66667E-01	1.00000E+00	-3.33333E-01
32	8.74372E-01	8.81081E-01	-6.70898E-03
33	8.82353E-01	8.28571E-01	5.37820E-02
34	7.40741E-01	9.04762E-01	-1.64021E-01
35	5.00000E-01	5.00000E-01	0.00000E+00
36	7.39130E-01	9.62963E-01	-2.23833E-01
37	6.00000E-01	6.66667E-01	-6.66670E-02
38	1.00000E+00	1.00000E+00	0.00000E+00
39	5.00000E-01	1.00000E+00	-5.00000E-01
40	7.77778E-01	1.00000E+00	-2.22222E-01
41	1.00000E+00	8.00000E-01	2.00000E-01
42	1.00000E+00	8.57143E-01	1.42857E-01
43	1.00000E+00	1.00000E+00	0.00000E+00
44	1.00000E+00	8.00000E-01	2.00000E-01
45	1.00000E+00	1.00000E+00	0.00000E+00
46	1.00000E+00	1.00000E+00	0.00000E+00
47	1.00000E+00	1.00000E+00	0.00000E+00
48	7.50000E-01	9.47368E-01	-1.97368E-01
49	1.00000E+00	1.00000E+00	0.00000E+00
50	1.00000E+00	7.50000E-01	2.50000E-01
51	1.00000E+00	1.00000E+00	0.00000E+00
52	9.00000E-01	5.00000E-01	4.00000E-01
53	1.00000E+00	1.00000E+00	0.00000E+00
54	1.00000E+00	1.00000E+00	0.00000E+00
55	1.00000E+00	1.00000E+00	0.00000E+00
56	1.00000E+00	1.00000E+00	0.00000E+00
57	1.00000E+00	1.00000E+00	0.00000E+00
58	1.00000E+00	1.00000E+00	0.00000E+00
59	1.00000E+00	1.00000E+00	0.00000E+00
60	1.00000E+00	1.00000E+00	0.00000E+00
61	1.00000E+00	1.00000E+00	0.00000E+00
62	1.00000E+00	1.00000E+00	0.00000E+00
63	1.00000E+00	1.00000E+00	0.00000E+00

69	1.00000E+00	5.71429E-01	4.28571E-01
70	6.66667E-01	1.00000E+00	-3.33333E-01
71	1.00000E+00	6.66667E-01	3.33333E-01
72	8.69565E-01	9.23077E-01	-5.35120E-02
73	1.00000E+00	1.00000E+00	0.00000E+00
74	4.00000E-01	1.00000E+00	-6.00000E-01
75	1.00000E+00	1.00000E+00	0.00000E+00
76	8.75000E-01	1.00000E+00	-1.25000E-01
77	1.00000E+00	1.00000E+00	0.00000E+00
78	1.00000E+00	1.00000E+00	0.00000E+00
79	1.00000E+00	1.00000E+00	0.00000E+00
80	9.54545E-01	9.25926E-01	2.86190E-02
81	8.00000E-01	1.00000E+00	-2.00000E-01
82	1.00000E+00	1.00000E+00	0.00000E+00
83	1.00000E+00	1.00000E+00	0.00000E+00
84	6.66667E-01	1.00000E+00	-3.33333E-01
85	1.00000E+00	1.00000E+00	0.00000E+00
86	1.00000E+00	1.00000E+00	0.00000E+00
87	1.00000E+00	1.00000E+00	0.00000E+00
88	1.00000E+00	5.00000E-01	5.00000E-01
89	1.00000E+00	1.00000E+00	0.00000E+00
90	1.00000E+00	1.00000E+00	0.00000E+00
91	1.00000E+00	1.00000E+00	0.00000E+00
92	8.33333E-01	1.00000E+00	-1.66667E-01
93	1.00000E+00	1.00000E+00	0.00000E+00
94	1.00000E+00	1.00000E+00	0.00000E+00
95	1.00000E+00	1.00000E+00	0.00000E+00
96	9.44444E-01	8.66667E-01	7.77770E-02
97	1.00000E+00	8.75000E-01	1.25000E-01
98	8.88889E-01	3.33333E-01	5.55556E-01
99	1.00000E+00	1.00000E+00	0.00000E+00
100	6.66667E-01	7.50000E-01	-8.33330E-02
101	1.00000E+00	1.00000E+00	0.00000E+00
102	1.00000E+00	1.00000E+00	0.00000E+00
103	1.00000E+00	1.00000E+00	0.00000E+00
104	1.00000E+00	8.00000E-01	2.00000E-01
105	1.00000E+00	1.00000E+00	0.00000E+00
106	1.00000E+00	1.00000E+00	0.00000E+00
107	1.00000E+00	1.00000E+00	0.00000E+00
108	1.00000E+00	1.00000E+00	0.00000E+00
109	1.00000E+00	1.00000E+00	0.00000E+00
110	1.00000E+00	1.00000E+00	0.00000E+00
111	1.00000E+00	1.00000E+00	0.00000E+00
112	8.60465E-01	7.87879E-01	7.25860E-02
113	7.50000E-01	1.00000E+00	-2.50000E-01
114	7.50000E-01	1.00000E+00	-2.50000E-01
115	1.00000E+00	1.00000E+00	0.00000E+00
116	7.50000E-01	1.00000E+00	-2.50000E-01
117	1.00000E+00	1.00000E+00	0.00000E+00
118	1.00000E+00	1.00000E+00	0.00000E+00
119	1.00000E+00	1.00000E+00	0.00000E+00
120	1.00000E+00	1.00000E+00	0.00000E+00
121	1.00000E+00	1.00000E+00	0.00000E+00
122	1.00000E+00	1.00000E+00	0.00000E+00
123	1.00000E+00	1.00000E+00	0.00000E+00
124	1.00000E+00	1.00000E+00	0.00000E+00
125	1.00000E+00	1.00000E+00	0.00000E+00
126	1.00000E+00	1.00000E+00	0.00000E+00
127	1.00000E+00	1.00000E+00	0.00000E+00
128	8.78173E-01	9.13793E-01	-3.56200E-02
129	8.05564E-01	9.23077E-01	-1.17521E-01

135	6.66667E-01	6.66667E-01	0.00000E+00
136	8.69565E-01	8.75000E-01	-5.43499E-03
137	6.66667E-01	6.66667E-01	0.00000E+00
138	1.00000E+00	1.00000E+00	0.00000E+00
139	1.00000E+00	1.00000E+00	0.00000E+00
140	1.00000E+00	5.00000E-01	5.00000E-01
141	1.00000E+00	1.00000E+00	0.00000E+00
142	1.00000E+00	1.00000E+00	0.00000E+00
143	1.00000E+00	1.00000E+00	0.00000E+00
144	9.16667E-01	8.18182E-01	9.84850E-02
145	8.00000E-01	1.00000E+00	-2.00000E-01
146	1.00000E+00	1.00000E+00	0.00000E+00
147	1.00000E+00	1.00000E+00	0.00000E+00
148	8.00000E-01	6.66667E-01	1.33333E-01
149	1.00000E+00	1.00000E+00	0.00000E+00
150	1.00000E+00	1.00000E+00	0.00000E+00
151	1.00000E+00	1.00000E+00	0.00000E+00
152	5.00000E-01	1.00000E+00	-5.00000E-01
153	1.00000E+00	1.00000E+00	0.00000E+00
154	1.00000E+00	1.00000E+00	0.00000E+00
155	1.00000E+00	1.00000E+00	0.00000E+00
156	1.00000E+00	1.00000E+00	0.00000E+00
157	1.00000E+00	1.00000E+00	0.00000E+00
158	1.00000E+00	1.00000E+00	0.00000E+00
159	1.00000E+00	1.00000E+00	0.00000E+00
160	9.50000E-01	9.00000E-01	5.00000E-02
161	5.00000E-01	1.00000E+00	-5.00000E-01
162	1.00000E+00	1.00000E+00	0.00000E+00
163	1.00000E+00	1.00000E+00	0.00000E+00
164	8.00000E-01	6.00000E-01	2.00000E-01
165	1.00000E+00	1.00000E+00	0.00000E+00
166	1.00000E+00	1.00000E+00	0.00000E+00
167	1.00000E+00	1.00000E+00	0.00000E+00
168	1.00000E+00	1.00000E+00	0.00000E+00
169	1.00000E+00	1.00000E+00	0.00000E+00
170	1.00000E+00	1.00000E+00	0.00000E+00
171	1.00000E+00	1.00000E+00	0.00000E+00
172	1.00000E+00	1.00000E+00	0.00000E+00
173	1.00000E+00	1.00000E+00	0.00000E+00
174	1.00000E+00	1.00000E+00	0.00000E+00
175	1.00000E+00	1.00000E+00	0.00000E+00
176	7.50000E-01	6.66667E-01	8.33330E-02
177	1.00000E+00	1.00000E+00	0.00000E+00
178	1.00000E+00	5.00000E-01	5.00000E-01
179	1.00000E+00	1.00000E+00	0.00000E+00
180	1.00000E+00	1.00000E+00	0.00000E+00
181	1.00000E+00	1.00000E+00	0.00000E+00
182	1.00000E+00	1.00000E+00	0.00000E+00
183	1.00000E+00	1.00000E+00	0.00000E+00
184	1.00000E+00	1.00000E+00	0.00000E+00
185	1.00000E+00	1.00000E+00	0.00000E+00
186	1.00000E+00	1.00000E+00	0.00000E+00
187	1.00000E+00	1.00000E+00	0.00000E+00
188	1.00000E+00	1.00000E+00	0.00000E+00
189	1.00000E+00	1.00000E+00	0.00000E+00
190	1.00000E+00	1.00000E+00	0.00000E+00
191	1.00000E+00	1.00000E+00	0.00000E+00
192	8.75000E-01	9.58333E-01	-8.33330E-02
193	8.78788E-01	8.15789E-01	6.29990E-02
194	1.00000E+00	1.00000E+00	0.00000E+00

198	1.00000E+00	5.00000E-01	5.00000E-01
199	5.00000E-01	6.66667E-01	-1.66667E-01
200	7.14286E-01	1.00000E+00	-2.85714E-01
201	1.00000E+00	6.00000E-01	4.00000E-01
202	1.00000E+00	1.00000E+00	0.00000E+00
203	1.00000E+00	1.00000E+00	0.00000E+00
204	1.00000E+00	1.00000E+00	0.00000E+00
205	1.00000E+00	1.00000E+00	0.00000E+00
206	1.00000E+00	1.00000E+00	0.00000E+00
207	1.00000E+00	1.00000E+00	0.00000E+00
208	5.00000E-01	1.00000E+00	-5.00000E-01
209	1.00000E+00	1.00000E+00	0.00000E+00
210	1.00000E+00	1.00000E+00	0.00000E+00
211	1.00000E+00	5.00000E-01	5.00000E-01
212	1.00000E+00	1.00000E+00	0.00000E+00
213	1.00000E+00	1.00000E+00	0.00000E+00
214	1.00000E+00	1.00000E+00	0.00000E+00
215	1.00000E+00	1.00000E+00	0.00000E+00
216	1.00000E+00	1.00000E+00	0.00000E+00
217	1.00000E+00	1.00000E+00	0.00000E+00
218	1.00000E+00	1.00000E+00	0.00000E+00
219	1.00000E+00	1.00000E+00	0.00000E+00
220	1.00000E+00	1.00000E+00	0.00000E+00
221	1.00000E+00	1.00000E+00	0.00000E+00
222	1.00000E+00	1.00000E+00	0.00000E+00
223	1.00000E+00	1.00000E+00	0.00000E+00
224	1.00000E+00	1.00000E+00	0.00000E+00
225	1.00000E+00	8.88889E-01	1.11111E-01
226	1.00000E+00	1.00000E+00	0.00000E+00
227	1.00000E+00	1.00000E+00	0.00000E+00
228	1.00000E+00	1.00000E+00	0.00000E+00
229	1.00000E+00	1.00000E+00	0.00000E+00
230	1.00000E+00	1.00000E+00	0.00000E+00
231	1.00000E+00	1.00000E+00	0.00000E+00
232	1.00000E+00	1.00000E+00	0.00000E+00
233	1.00000E+00	1.00000E+00	0.00000E+00
234	1.00000E+00	1.00000E+00	0.00000E+00
235	1.00000E+00	1.00000E+00	0.00000E+00
236	1.00000E+00	1.00000E+00	0.00000E+00
237	1.00000E+00	1.00000E+00	0.00000E+00
238	1.00000E+00	1.00000E+00	0.00000E+00
239	1.00000E+00	1.00000E+00	0.00000E+00
240	1.00000E+00	1.00000E+00	0.00000E+00
241	1.00000E+00	6.66667E-01	3.33333E-01
242	1.00000E+00	1.00000E+00	0.00000E+00
243	1.00000E+00	1.00000E+00	0.00000E+00
244	1.00000E+00	1.00000E+00	0.00000E+00
245	1.00000E+00	1.00000E+00	0.00000E+00
246	1.00000E+00	1.00000E+00	0.00000E+00
247	1.00000E+00	1.00000E+00	0.00000E+00
248	1.00000E+00	1.00000E+00	0.00000E+00
249	1.00000E+00	1.00000E+00	0.00000E+00
250	1.00000E+00	1.00000E+00	0.00000E+00
251	1.00000E+00	1.00000E+00	0.00000E+00
252	1.00000E+00	1.00000E+00	0.00000E+00
253	1.00000E+00	1.00000E+00	0.00000E+00
254	1.00000E+00	1.00000E+00	0.00000E+00
255	1.00000E+00	1.00000E+00	0.00000E+00